

RK611  
RK06, RK07

RK611 DSKLS PRT4  
CZR6DC0

AH-9110C-MC  
FICHE 1 OF 1

NOV 1980  
COPYRIGHT © 76-80  
MADE IN USA



A large grid of data tables, likely a microfiche or microfilm page, containing multiple columns and rows of text and numbers. The content is too small to read clearly but appears to be organized in a structured format. A small barcode is visible in the bottom right corner of the grid area.

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30

.REM % IDENTIFICATION

PRODUCT CODE: AC-9108C-MC  
PRODUCT NAME: CZR6DCO RK611 DSKLS PRT4  
DATE: JUNE,1980  
MAINTAINER: DIAGNOSTIC GROUP

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.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1976,1980 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

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

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

56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111

## 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

112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167

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

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

IF A ^S HAS BEEN TYPED, TYPING THE ^Q CANCELS THE STALL INITIATED BY THE ^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

224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279

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.

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

\*\*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 DAT LATE 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 CONTROLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF ON 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 RECOGNI

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.



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

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

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

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

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

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 FOLLOWING 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

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

672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727

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

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 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

NOTE: SINCE 18 BIT FUNCTIONALITY OF THE RK611 IS NOT  
UTILIZED ON THE PDP-11, THE FOLLOWING TESTS, INTENDED  
FOR MANUFACTURING USE, WILL NOT BE RUN UNLESS LOCATION  
"SEC24" IS PATCHED TO A NON-ZERO VALUE.

TEST 46 18 BIT WRITE DATA (PART 1)



784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839

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.  
VERIFY THAT TWO ZERO BITS ARE PRESENT FOR EACH WORD.

TEST 47 18 BIT WRITE DATA (PART 2)

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	WWWWW	AAAAA	

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

840	POSTAMBLE	20	1
841	GAP		
842	(22(10) SECTOR/TRACK)	160	7
843	(20(10) SECTOR/TRACK)	140	6

844  
845 REFER TO THE RK06 UNIBUS DISK SUBSYSTEM SPECIFICATION FOR MORE  
846 DETAILED DESCRIPTION.

847  
848 THE 'BIT COUNT' REPORTED IS INITIALIZED AT THE START OF EACH FIELD  
849 AND IS INCREMENTED FOR EACH BIT PROCESSED.

850  
851 WHEN THE OPERATION BEING PERFORMED INVOLVES WRITING, OTHER PERTINENT  
852 INFORMATION MAY CONTAIN WORDS LABELED PRESENT BIT, PRESENT BIT -1,  
853 PRESENT BIT -2, AND PRESENT BIT +1. THESE BITS ARE PRESENTED IN THIS  
854 WAY TO SHOW THE WRITE DATA STREAM IN THE SAME MANNER THAT THE  
855 RK611 LOOKS AT THE DATA STREAM TO COMPUTE PRECOMPENSATION ADVANCE  
856 AND PRECOMPENSATION DELAY IN THE WRITE OPERATION.

857  
858 WHEN THE OPERATION BEING PERFORMED INVOLVES READING, OTHER PERTINENT  
859 INFORMATION MAY CONTAIN PREVIOUS BIT AND PRESENT BIT WORDS. THESE  
860 BITS RELATE TO RK611 LOGIC THAT DETERMINES WHEN THE BIT IS VALID  
861 FROM THE DECODER.

862

z

```
863      ; *** REV 003 ***
864      .NLIST  CND,MD,MC
865      .LIST   ME
866      .ENABL  ABS,AMA
867      $SWR=  167400
868      $TN=    1
869      .TITLE  CZR6DC0 RK611 DSKLS PRT4
870      ;*COPYRIGHT (C) 1976,1980
871      ;*DIGITAL EQUIPMENT CORP.
872      ;*MAYNARD, MASS. 01754
873      ;*
874      ;*
875      ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
876      ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
877      ;*
878      .SBTTL  OPERATIONAL SWITCH SETTINGS
879      ;*
880      ;*      SWITCH          USE
881      ;*      -----          -----
882      ;*      15             HALT ON ERROR
883      ;*      14             LOOP ON TEST
884      ;*      13             INHIBIT ERROR TYPEOUTS
885      ;*      12             ABORT PROGRAM AFTER 20 ERRORS
886      ;*      11             INHIBIT ITERATIONS
887      ;*      10             BELL ON ERROR
888      ;*      9              LOOP ON ERROR
889      ;*      8              LOOP ON TEST IN SWR<7:0>
890      .SBTTL  BASIC DEFINITIONS
891      ;*
892      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
893      STACK= 1100
894      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
895      .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
896      ;*
897      ;*MISCELLANEOUS DEFINITIONS
898      HT=    11              ;;CODE FOR HORIZONTAL TAB
899      LF=    12              ;;CODE FOR LINE FEED
900      CR=    15              ;;CODE FOR CARRIAGE RETURN
901      CRLF=  200            ;;CODE FOR CARRIAGE RETURN-LINE FEED
902      PS=    177776         ;;PROCESSOR STATUS WORD
903      .EQUIV  PS,PSW
904      STKLMT= 177774         ;;STACK LIMIT REGISTER
905      PIRQ=   177772         ;;PROGRAM INTERRUPT REQUEST REGISTER
906      DSWR=   177570         ;;HARDWARE SWITCH REGISTER
907      DDISP=  177570         ;;HARDWARE DISPLAY REGISTER
908      ;*
909      ;*GENERAL PURPOSE REGISTER DEFINITIONS
910      R0=    %0              ;;GENERAL REGISTER
911      R1=    %1              ;;GENERAL REGISTER
912      R2=    %2              ;;GENERAL REGISTER
913      R3=    %3              ;;GENERAL REGISTER
914      R4=    %4              ;;GENERAL REGISTER
915      R5=    %5              ;;GENERAL REGISTER
916      R6=    %6              ;;GENERAL REGISTER
917      R7=    %7              ;;GENERAL REGISTER
918      SP=    %6              ;;STACK POINTER
```

```
919          000007          PC=      7          ;;PROGRAM COUNTER
920
921          ;*PRIORITY LEVEL DEFINITIONS
922          000000          PR0=      0          ;;PRIORITY LEVEL 0
923          000040          PR1=     40          ;;PRIORITY LEVEL 1
924          000100          PR2=    100          ;;PRIORITY LEVEL 2
925          000140          PR3=    140          ;;PRIORITY LEVEL 3
926          000200          PR4=    200          ;;PRIORITY LEVEL 4
927          000240          PR5=    240          ;;PRIORITY LEVEL 5
928          000300          PR6=    300          ;;PRIORITY LEVEL 6
929          000340          PR7=    340          ;;PRIORITY LEVEL 7
930
931          ;*'SWITCH REGISTER' SWITCH DEFINITIONS
932          100000          SW15=   100000
933          040000          SW14=    40000
934          020000          SW13=    20000
935          010000          SW12=    10000
936          004000          SW11=     4000
937          002000          SW10=     2000
938          001000          SW09=     1000
939          000400          SW08=      400
940          000200          SW07=      200
941          000100          SW06=      100
942          000040          SW05=       40
943          000020          SW04=       20
944          000010          SW03=       10
945          000004          SW02=        4
946          000002          SW01=        2
947          000001          SW00=        1
948          .EQUIV          SW09,SW9
949          .EQUIV          SW08,SW8
950          .EQUIV          SW07,SW7
951          .EQUIV          SW06,SW6
952          .EQUIV          SW05,SW5
953          .EQUIV          SW04,SW4
954          .EQUIV          SW03,SW3
955          .EQUIV          SW02,SW2
956          .EQUIV          SW01,SW1
957          .EQUIV          SW00,SW0
958
959          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
960          100000          BIT15=  100000
961          040000          BIT14=   40000
962          020000          BIT13=   20000
963          010000          BIT12=   10000
964          004000          BIT11=    4000
965          002000          BIT10=    2000
966          001000          BIT09=    1000
967          000400          BIT08=     400
968          000200          BIT07=     200
969          000100          BIT06=     100
970          000040          BIT05=      40
971          000020          BIT04=      20
972          000010          BIT03=      10
973          000004          BIT02=       4
974          000002          BIT01=       2
```

```
975          000001          BIT00= 1
976          .EQUIV BIT09,BIT9
977          .EQUIV BIT08,BIT8
978          .EQUIV BIT07,BIT7
979          .EQUIV BIT06,BIT6
980          .EQUIV BIT05,BIT5
981          .EQUIV BIT04,BIT4
982          .EQUIV BIT03,BIT3
983          .EQUIV BIT02,BIT2
984          .EQUIV BIT01,BIT1
985          .EQUIV BIT00,BIT0
986
987          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
988          000004          ERRVEC= 4          ;; TIME OUT AND OTHER ERRORS
989          000010          RESVEC= 10         ;; RESERVED AND ILLEGAL INSTRUCTIONS
990          000014          TBITVEC=14        ;; "T" BIT
991          000014          TRTVEC= 14         ;; TRACE TRAP
992          000014          BPTVEC= 14         ;; BREAKPOINT TRAP (BPT)
993          000020          IOTVEC= 20         ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
994          000024          PWRVEC= 24         ;; POWER FAIL
995          000030          EMTVEC= 30         ;; EMULATOR TRAP (EMT) **ERROR**
996          000034          TRAPVEC=34        ;; "TRAP" TRAP
997          000060          TKVEC= 60          ;; TTY KEYBOARD VECTOR
998          000064          TPVEC= 64          ;; TTY PRINTER VECTOR
999          000240          PIRQVEC=240       ;; PROGRAM INTERRUPT REQUEST VECTOR
1000          000210          AVECT1= 210      ;DEFINE RK611 VECTOR ADDRESS
1001          000240          APRIOR= PR5      ;DEFINE RK611 PRIORITY
1002          177440          ABASE= 177440    ;DEFINE BASE OF RK611 REGISTERS
1003
1004          .SBTTL RK611 CONTROLLER REGISTER DEFINITION
1005
1006          000000          RKCS1= 0          ;CONTROL AND STATUS REGISTER 1
1007          000002          RKWC= 2          ;WORD COUNT REGISTER
1008          000004          RKBA= 4          ;BUS ADDRESS REGISTER
1009          000006          RKDA= 6          ;DESIRED TRACK SECTOR REGISTER
1010          000010          RKCS2= 10        ;CONTROL AND STATUS REGISTER 2
1011          000012          RKDS= 12        ;DRIVE STATUS REGISTER
1012          000014          RKER= 14        ;ERROR REGISTER
1013          000016          RKASOF= 16       ;ATTENTION SUMMARY AND OFFSET REGISTER
1014          000020          RKDCYL= 20       ;DESIRED CYLINDER REGISTER
1015          000024          RKDB= 24        ;DATA BUFFER
1016          000026          RKMR1= 26       ;MAINTENANCE REGISTER 1
1017          000034          RKMR2= 34       ;MAINTENANCE REGISTER 2
1018          000036          RKMR3= 36       ;MAINTENANCE REGISTER 3
1019          000030          RKECPS= 30      ;ECC POSITION INFORMATION
1020          000032          RKECPT= 32      ;ECC PATTERN INFORMATION
1021          000022          RKSPAR= 22      ;SPARE REGISTER
1022
1023          .SBTTL DRIVE COMMANDS
1024
1025          000001          SELDRV= 01       ;SELECT DRIVE
1026          000003          PACK= 03        ;PACK ACKNOWLEDGE
1027          000005          CLEAR= 05       ;DRIVE CLEAR
1028          000007          UNLOAD= 07      ;UNLOAD
1029          000011          SRTSPL= 11      ;START SPINDLE
1030          000013          RECAL= 13       ;RECALIBRATE
```

1031	000015	OFFSET= 15	:OFFSET
1032	000017	SEEK= 17	:SEEK
1033	000021	RDDATA= 21	:READ DATA
1034	000023	WRDATA= 23	:WRITE DATA
1035	000025	RDHEAD= 25	:READ HEADER
1036	000027	WRHEAD= 27	:WRITE HEADER AND DATA
1037	000031	WRTCHK= 31	:WRITE CHECK
1038	000300	INTR= 300	:GENERATE INTERRUPT TO CPU
1039			
1040		.SBTTL CONTROL AND STATUS REGISTER 1 BITS	
1041			
1042	000001	GO= BIT0	:GO BIT
1043	000100	IE= BIT6	:INTERRUPT ENABLE
1044	000200	RDY= BIT7	:CONTROLLER READY
1045	000400	BA16= BIT8	:BUS ADDRESS BIT 16
1046	001000	BA17= BIT9	:BUS ADDRESS BIT 17
1047	002000	CDT= BIT10	:CONTROLLER DRIVE TYPE (0=RK06)
1048	004000	CTO= BIT11	:CONTROLLER TIMED OUT WAITING FOR : DRIVE RESPONSE
1049			
1050	010000	CFMT= BIT12	:CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
1051	020000	SPAR= BIT13	:DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1052	040000	DI= BIT14	:DRIVE INTERRUPT
1053	100000	CERR= BIT15	:CONTROLLER ERROR
1054	100000	CCLR= BIT15	:CONTROLLER CLEAR
1055			
1056		.SBTTL CONTROL AND STATUS REGISTER 2 BITS	
1057			
1058	000007	DRVMSK= 7	:MASK FOR DRIVE SELECTION CODE
1059	000010	RLS= BIT3	:DESELECT OR RELEASE DRIVE IN BITS 0-2
1060	000020	BAI= BIT4	:BUS ADDRESS INCREMENT INHIBIT
1061	000040	SCLR= BIT5	:CLEAR CONTROLLER AND ALL DRIVES
1062	000100	IR= BIT6	:INPUT READY
1063	000200	OR= BIT7	:OUTPUT READY
1064	000400	UFE= BIT8	:UNIT FIELD ERROR
1065	001000	MDS= BIT9	:MULTIPLE DRIVE SELECT
1066	002000	PGE= BIT10	:PROGRAMMING ERROR
1067	004000	NEM= BIT11	:NON-EXISTENT MEMORY
1068	010000	NED= BIT12	:NON-EXISTENT DRIVE
1069	020000	UPE= BIT13	:UNIBUS PARITY ERROR
1070	040000	WCE= BIT14	:WRITE CHECK ERROR
1071	100000	DLT= BIT15	:DATA LATE ERROR
1072			
1073		.SBTTL ERROR REGISTER BIT DEFINITION	
1074			
1075	000001	ILF= BIT0	:ILLEGAL FUNCTION CODE
1076	000002	SKI= BIT1	:SEEK INCOMPLETE
1077	000004	NXF= BIT2	:NON-EXECUTABLE DRIVE FUNCTION
1078	000010	DRPAR= BIT3	:DRIVE DETECTED DRIVE BUS PARITY ERROR
1079	000020	FMTE= BIT4	:FORMAT ERROR
1080	000040	DTYPE= BIT5	:DRIVE TYPE ERROR
1081	000100	ECH= BIT6	:ECC HARD
1082	000200	BSE= BIT7	:BAD SECTOR ERROR
1083	000400	HVRC= BIT8	:HEADER VRC ERROR
1084	001000	COE= BIT9	:CYLINDER ADDRESS OVERFLOW ERROR
1085	002000	IDAE= BIT10	:INVALID DISK ADDRESS ERROR
1086	004000	WLE= BIT11	:WRITE LOCK ERROR

```
1087      010000      DTE=   BIT12      ;DRIVE TIMING ERROR
1088      020000      OPI=   BIT13      ;OPERATION (SEARCH) INCOMPLETE
1089      040000      UNS=   BIT14      ;DRIVE UNSAFE
1090      100000      DCK=   BIT15      ;DATA CHECK
1091
1092      .SBTTL  STATUS REGISTER BIT DEFINITION
1093
1094      000001      DRA=   BIT0       ;DRIVE AVAILABLE (CONTROLLER IS SET IF
1095      ; THIS BIT IS RESET)
1096      000004      OFST=  BIT2       ;DRIVE OFFSET
1097      000010      ACLO=  BIT3       ;AC LOW
1098      000020      SODLSS= BIT4      ;SPEED LOSS
1099      000040      DROT=  BIT5       ;DRIVE OFF TRACK
1100      000100      VV=    BIT6       ;VOLUME VALID
1101      000200      DRDY=  BIT7       ;DRIVE READY
1102      000400      DDT=   BIT8       ;DRIVE TYPE (0=RK06)
1103      004000      WRL=   BIT11      ;WRITE LOCK
1104      020000      PIP=   BIT13      ;POSITIONING IN PROGRESS
1105      040000      DSC=   BIT14      ;DRIVE STATUS CHANGE
1106      100000      SVAL=  BIT15      ;STATUS VALID
1107
1108      .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
1109
1110      000017      MESMSK= 17      ;MESSAGE MASK
1111
1112      000020      PAT=   BIT4       ;FORCE EVEN PARITY ON DRIVE MESSAGE LINES
1113      000040      DMD=   BIT5       ;DIAGNOSTIC MODE
1114      000100      MSP=   BIT6       ;MAINTENANCE SECTOR PULSE
1115      000200      MIND=  BIT7       ;MAINTENANCE INDEX
1116      000400      MCLK=  BIT8       ;MAINTENANCE CLOCK
1117      001000      MERD=  BIT9       ;MAINTENANCE ENCODED READ DATA
1118      002000      MEWD=  BIT10      ;MAINTENANCE ENCODED WRITE DATA
1119      004000      PCA=   BIT11      ;PRECOMPENSATION ADVANCE
1120      010000      PCD=   BIT12      ;PRECOMPENSATION DELAY
1121      020000      ECCW=  BIT13      ;ECC WORD IS BEING READ OR WRITTEN
1122      040000      WRTGAT= BIT14     ;WRITE GATE
1123      100000      RDGATE= BIT15     ;READ GATE
1124
1125      .SBTTL  TRANSMITTED MESSAGE A
1126
1127      000020      S.SEK=  BIT4       ;SEEK COMMAND
1128      000040      S.RECL= BIT5       ;RECALIBRATE COMMAND
1129      000100      S.STSP= BIT6       ;START SPINDLE COMMAND
1130      000200      S.RTC=  BIT7       ;DRIVE RETURN TO CENTERLINE COMMAND
1131      000400      S.CLR=  BIT8       ;CLEAR ERROR AND DSC
1132      001000      S.FMT=  BIT9       ;FORMAT
1133      002000      S.UNLD= BIT10      ;UNLOAD
1134      004000      S.PACK= BIT11      ;SET VOLUME VALID (PACK ACKNOWLEDGE)
1135      .SBTTL  TRAP CATCHER
1136
1137      000000      .=0
1138      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1139      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1140      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1141      000174      .=174
1142      000174      000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
```

```
1143 000176 000000 SWREG: .WORD 0 ;;SOFTWARE SWITCH REGISTER
1144 .SBTTL STARTING ADDRESS(ES)
1145 000200 000137 003336 JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1146 000204 000137 003326 JMP RESTRT ;;JUMP TO RESTART ROUTINE
1147 .=214
1148 000214 000137 003316 JMP PARM ;;JUMP TO OPERATOR ASSIGNED PARAMETERS
1149 .SBTTL ACT11 HOOKS
1150
1151 ;;*****
1152 ;HOOKS REQUIRED BY ACT11
1153 $SVPC=. ;SAVE PC
1154 .=46
1155 000046 033150 $ENDAD ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1156 .=52
1157 000052 000000 .WORD 0 ;;2)SET LOC.52 TO ZERO
1158 .=$SVPC ;; RESTORE PC
1159 .=1000
1160 .SBTTL APT PARAMETER BLOCK
1161
1162 ;;*****
1163 ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1164 ;;*****
1165 .SX=. ;;SAVE CURRENT LOCATION
1166 .=24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1167 000024 000200 200 ;;FOR APT START UP
1168 .=44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
1169 000044 001000 $APTHDR ;;POINT TO APT HEADER BLOCK
1170 .=$X ;;RESET LOCATION COUNTER
1171 ;;*****
1172 ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1173 ;INTERFACE SPEC.
1174
1175 $APTHD:
1176 001000 000000 $HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1177 001002 001214 $MBADR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1178 001004 000000 $STSM: .WORD ;;RUN TIM OF LONGEST TEST
1179 001006 000000 $PASTM: .WORD ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1180 001010 000000 $UNITM: .WORD ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1181 001012 000032 .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```



```
1182      .SBTTL COMMON TAGS
1183
1184      ;:*****
1185      ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1186      ;*USED IN THE PROGRAM.
1187
1188      001100      .=1100
1189      001100      SCMTAG:      ;;START OF COMMON TAGS
1190      001100      000000      .WORD      0
1191      001102      000      $TSTNM: .BYTE      0      ;;CONTAINS THE TEST NUMBER
1192      001103      000      $ERFLG: .BYTE      0      ;;CONTAINS ERROR FLAG
1193      001104      000000      $!CNT: .WORD      0      ;;CONTAINS SUBTEST ITERATION COUNT
1194      001106      000000      $LPADR: .WORD      0      ;;CONTAINS SCOPE LOOP ADDRESS
1195      001110      000000      $LPERR: .WORD      0      ;;CONTAINS SCOPE RETURN FOR ERRORS
1196      001112      000000      $ERTTL: .WORD      0      ;;CONTAINS TOTAL ERRORS DETECTED
1197      001114      000      $ITEMB: .BYTE      0      ;;CONTAINS ITEM CONTROL BYTE
1198      001115      001      $ERMAX: .BYTE      1      ;;CONTAINS MAX. ERRORS PER TEST
1199      001116      000000      $ERRPC: .WORD      0      ;;CONTAINS PC OF LAST ERROR INSTRUCTION
1200      001120      000000      $GDADR: .WORD      0      ;;CONTAINS ADDRESS OF 'GOOD' DATA
1201      001122      000000      $BDADR: .WORD      0      ;;CONTAINS ADDRESS OF 'BAD' DATA
1202      001124      000000      $GDDAT: .WORD      0      ;;CONTAINS 'GOOD' DATA
1203      001126      000000      $BDDAT: .WORD      0      ;;CONTAINS 'BAD' DATA
1204      001130      000000      .WORD      0      ;;RESERVED--NOT TO BE USED
1205      001132      000000      .WORD      0
1206      001134      000      $AUTOB: .BYTE      0      ;;AUTOMATIC MODE INDICATOR
1207      001135      000      $INTAG: .BYTE      0      ;;INTERRUPT MODE INDICATOR
1208      001136      000000      .WORD      0
1209      001140      177570      SWR:      .WORD      DSWR      ;;ADDRESS OF SWITCH REGISTER
1210      001142      177570      DISPLAY: .WORD      DDISP      ;;ADDRESS OF DISPLAY REGISTER
1211      001144      177560      $TKS:      177560      ;;TTY KBD STATUS
1212      001146      177562      $TKB:      177562      ;;TTY KBD BUFFER
1213      001150      177564      $TPS:      177564      ;;TTY PRINTER STATUS REG. ADDRESS
1214      001152      177566      $TPB:      177566      ;;TTY PRINTER BUFFER REG. ADDRESS
1215      001154      000      $NULL: .BYTE      0      ;;CONTAINS NULL CHARACTER FOR FILLS
1216      001155      002      $FILLS: .BYTE      2      ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
1217      001156      012      $FILLC: .BYTE      12      ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
1218      001157      000      $TPFLG: .BYTE      0      ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1219      001160      000000      $TMP0: .WORD      0      ;;USER DEFINED
1220      001162      000000      $TMP1: .WORD      0      ;;USER DEFINED
1221      001164      000000      $TMP2: .WORD      0      ;;USER DEFINED
1222      001166      000000      $TMP3: .WORD      0      ;;USER DEFINED
1223      001170      000000      $TMP4: .WORD      0      ;;USER DEFINED
1224      001172      000000      $TMP5: .WORD      0      ;;USER DEFINED
1225      001174      000000      $TMP6: .WORD      0      ;;USER DEFINED
1226      001176      000000      $TMP7: .WORD      0      ;;USER DEFINED
1227      001200      000000      $TIMES: 0      ;;MAX. NUMBER OF ITERATIONS
1228      001202      000000      $ESCAPE: 0      ;;ESCAPE ON ERROR ADDRESS
1229      001204      177607      $BELL: .ASCIZ <207><377><377>      ;;CODE FOR BELL
1230      001210      077      $QUES: .ASCII  /?/      ;;QUESTION MARK
1231      001211      015      $CRLF: .ASCII  <15>      ;;CARRIAGE RETURN
1232      001212      000012      $LF: .ASCIZ  <12>      ;;LINE FEED
1233      ;:*****
1234      .SBTTL APT MAILBOX-ETABLE
1235
1236      ;:*****
1237      .EVEN
```

1238	001214		\$MAIL:		::APT MAILBOX
1239	001214	000000	\$MSGTY:	.WORD	AMSGTY ::MESSAGE TYPE CODE
1240	001216	000000	\$FATAL:	.WORD	AFATAL ::FATAL ERROR NUMBER
1241	001220	000000	\$TESTN:	.WORD	ATESTN ::TEST NUMBER
1242	001222	000000	\$PASS:	.WORD	APASS ::PASS COUNT
1243	001224	000000	\$DEVCT:	.WORD	ADEVCT ::DEVICE COUNT
1244	001226	000000	\$UNIT:	.WORD	AUNIT ::I/O UNIT NUMBER
1245	001230	000000	\$MSGAD:	.WORD	AMSGAD ::MESSAGE ADDRESS
1246	001232	000000	\$MSGLG:	.WORD	AMSGLG ::MESSAGE LENGTH
1247	001234		\$ETABLE:		::APT ENVIRONMENT TABLE
1248	001234	000	\$ENV:	.BYTE	AENV ::ENVIRONMENT BYTE
1249	001235	000	\$ENVM:	.BYTE	AENVM ::ENVIRONMENT MODE BITS
1250	001236	000000	\$SWREG:	.WORD	ASWREG ::APT SWITCH REGISTER
1251	001240	000000	\$USWR:	.WORD	AUSWR ::USER SWITCHES
1252	001242	000000	\$CPUOP:	.WORD	ACPUOP ::CPU TYPE,OPTIONS
1253			::*		BITS 15-11=CPU TYPE
1254			::*		11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1255			::*		11/70=06,PDQ=07,Q=10
1256			::*		BIT 10=REAL TIME CLOCK
1257			::*		BIT 9=FLOATING POINT PROCESSOR
1258			::*		BIT 8=MEMORY MANAGEMENT
1259	001244	000	\$MAMS1:	.BYTE	AMAMS1 ::HIGH ADDRESS,M.S. BYTE
1260	001245	000	\$MTYP1:	.BYTE	AMTYP1 ::MEM. TYPE,BLK#1
1261			::*		MEM.TYPE BYTE -- (HIGH BYTE)
1262			::*		900 NSEC CORE=001
1263			::*		300 NSEC BIPOLAR=002
1264			::*		500 NSEC MOS=003
1265	001246	000000	\$MADR1:	.WORD	AMADR1 ::HIGH ADDRESS,BLK#1
1266			::*		MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1267	001250	000	\$MAMS2:	.BYTE	AMAMS2 ::HIGH ADDRESS,M.S. BYTE
1268	001251	000	\$MTYP2:	.BYTE	AMTYP2 ::MEM.TYPE,BLK#2
1269	001252	000000	\$MADR2:	.WORD	AMADR2 ::MEM.LAST ADDRESS,BLK#2
1270	001254	000	\$MAMS3:	.BYTE	AMAMS3 ::HIGH ADDRESS,M.S.BYTE
1271	001255	000	\$MTYP3:	.BYTE	AMTYP3 ::MEM.TYPE,BLK#3
1272	001256	000000	\$MADR3:	.WORD	AMADR3 ::MEM.LAST ADDRESS,BLK#3
1273	001260	000	\$MAMS4:	.BYTE	AMAMS4 ::HIGH ADDRESS,M.S.BYTE
1274	001261	000	\$MTYP4:	.BYTE	AMTYP4 ::MEM.TYPE,BLK#4
1275	001262	000000	\$MADR4:	.WORD	AMADR4 ::MEM.LAST ADDRESS,BLK#4
1276	001264	000210	\$VECT1:	.WORD	AVECT1 ::INTERRUPT VECTOR#1,BUS PRIORITY#1
1277	001266	000000	\$VECT2:	.WORD	AVECT2 ::INTERRUPT VECTOR#2BUS PRIORITY#2
1278	001270	177440	\$BASE:	.WORD	ABASE ::BASE ADDRESS OF EQUIPMENT UNDER TEST
1279	001272	000000	\$DEVN:	.WORD	ADEVN ::DEVICE MAP
1280	001274	000000	\$CDW1:	.WORD	ACDW1 ::CONTROLLER DESCRIPTION WORD#1
1281	001276	000000	\$CDW2:	.WORD	ACDW2 ::CONTROLLER DESCRIPTION WORD#2
1282	001300		\$ETEND:		
1283			.MEXIT		

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

001300  
001300 044501  
001302 042506  
001304 041046  
001306 041402  
001310 000000  
001312 000000  
001314 041052  
001316 041406  
001320 044546  
001322 050442  
001324 041074  
001326 041432  
001330 044546  
001332 050460  
001334 041074  
001336 041432  
001340 044546  
001342 050501  
001344 041074  
001346 041432  
001350 044623  
001352 050442  
001354 041074  
001356 041432  
001360 044623  
001362 050460  
001364 041074  
001366 041432

.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 ENABLE 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

1340			:	MESS B INCORRECT
1341	001370	044623		EM301
1342	001372	050501		EM4002
1343	001374	041074		DT003
1344	001376	041432		DF003
1345			:	ERROR 11: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1346			:	CS1 INCORRECT
1347	001400	044701		EM302
1348	001402	050442		EM4000
1349	001404	041074		DT003
1350	001406	041432		DF003
1351			:	ERROR 12: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1352			:	MESS A INCORRECT
1353	001410	044701		EM302
1354	001412	050460		EM4001
1355	001414	041074		DT003
1356	001416	041432		DF003
1357			:	ERROR 13: ATTEMPTING TO CHECK SEEK MESSAGE FOR WRITE CHECK
1358			:	MESS B INCORRECT
1359	001420	044701		EM302
1360	001422	050501		EM4002
1361	001424	041074		DT003
1362	001426	041432		DF003
1363			:	ERROR 14: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1364			:	CS1 INCORRECT
1365	001430	044760		EM303
1366	001432	050442		EM4000
1367	001434	041074		DT003
1368	001436	041432		DF003
1369			:	ERROR 15: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1370			:	MESS A INCORRECT
1371	001440	044760		EM303
1372	001442	050460		EM4001
1373	001444	041074		DT003
1374	001446	041432		DF003
1375			:	ERROR 16: ATTEMPTING TO CHECK CLEAR MESSAGE FOR READ DATA
1376			:	MESS B INCORRECT
1377	001450	044760		EM303
1378	001452	050501		EM4002
1379	001454	041074		DT003
1380	001456	041432		DF003
1381			:	ERROR 17: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1382			:	CS1 INCORRECT
1383	001460	045036		EM304
1384	001462	050442		EM4000
1385	001464	041074		DT003
1386	001466	041432		DF003
1387			:	ERROR 20: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1388			:	MESS A INCORRECT
1389	001470	045036		EM304
1390	001472	050460		EM4001
1391	001474	041074		DT003
1392	001476	041432		DF003
1393			:	ERROR 21: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE DATA
1394			:	MESS B INCORRECT
1395	001500	045036		EM304

1396	001502	050501	EM4002
1397	001504	041074	DT003
1398	001506	041432	DF003
1399	:	:	ERROR 22: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1400	:	:	CS1 INCORRECT
1401	001510	045115	EM305
1402	001512	050442	EM4000
1403	001514	041074	DT003
1404	001516	041432	DF003
1405	:	:	ERROR 23: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1406	:	:	MESS A INCORRECT
1407	001520	045115	EM305
1408	001522	050460	EM4001
1409	001524	041074	DT003
1410	001526	041432	DF003
1411	:	:	ERROR 24: ATTEMPTING TO CHECK CLEAR MESSAGE FOR WRITE CHECK
1412	:	:	MESS B INCORRECT
1413	001530	045115	EM305
1414	001532	050501	EM4002
1415	001534	041074	DT003
1416	001536	041432	DF003
1417	:	:	ERROR 25: ATTEMPTING TO CHECK HEADER GENERATION
1418	:	:	WITH VARIOUS CYLINDER VALUES
1419	:	:	CS1 INCORRECT
1420	001540	045175	EM306
1421	001542	050442	EM4000
1422	001544	041074	DT003
1423	001546	041456	DF025
1424	:	:	ERROR 26: ATTEMPTING TO CHECK HEADER GENERATION
1425	:	:	WITH VARIOUS CYLINDER VALUES
1426	:	:	FIRST WORD OF HEADER INCORRECT
1427	001550	045175	EM306
1428	001552	050522	EM4003
1429	001554	041074	DT003
1430	001556	041456	DF025
1431	:	:	ERROR 27: ATTEMPTING TO CHECK HEADER GENERATION
1432	:	:	WITH VARIOUS CYLINDER VALUES
1433	:	:	SECOND WORD OF HEADER INCORRECT
1434	001560	045175	EM306
1435	001562	050552	EM4004
1436	001564	041074	DT003
1437	001566	041456	DF025
1438	:	:	ERROR 30: ATTEMPTING TO CHECK HEADER GENERATION
1439	:	:	WITH VARIOUS TRACK VALUES
1440	:	:	CS1 INCORRECT
1441	001570	045277	EM307
1442	001572	050442	EM4000
1443	001574	041074	DT003
1444	001576	041456	DF025
1445	:	:	ERROR 31: ATTEMPTING TO CHECK HEADER GENERATION
1446	:	:	WITH VARIOUS TRACK VALUES
1447	:	:	FIRST WORD OF HEADER INCORRECT
1448	001600	045277	EM307
1449	001602	050522	EM4003
1450	001604	041074	DT003
1451	001606	041456	DF025

1452	:	ERROR 32: ATTEMPTING TO CHECK HEADER GENERATION
1453	:	WITH VARIOUS TRACK VALUES
1454	:	SECOND WORD OF HEADER INCORRECT
1455	001610 045277	EM307
1456	001612 050552	EM4004
1457	001614 041074	DT003
1458	001616 041456	DF025
1459	:	ERROR 33: ATTEMPTING TO CHECK HEADER GENERATION
1460	:	WITH VARIOUS SECTOR VALUES
1461	:	CS1 INCORRECT
1462	001620 045376	EM308
1463	001622 050442	EM4000
1464	001624 041074	DT003
1465	001626 041456	DF025
1466	:	ERROR 34: ATTEMPTING TO CHECK HEADER GENERATION
1467	:	WITH VARIOUS SECTOR VALUES
1468	:	FIRST WORD OF HEADER INCORRECT
1469	001630 045376	EM308
1470	001632 050522	EM4003
1471	001634 041074	DT003
1472	001636 041456	DF025
1473	:	ERROR 35: ATTEMPTING TO CHECK HEADER GENERATION
1474	:	WITH VARIOUS SECTOR VALUES
1475	:	SECOND WORD OF HEADER INCORRECT
1476	001640 045376	EM308
1477	001642 050552	EM4004
1478	001644 041074	DT003
1479	001646 041456	DF025
1480	:	ERROR 36: ATTEMPTING TO CHECK HEADER GENERATION
1481	:	WITH VARIOUS FORMAT VALUES
1482	:	CS1 INCORRECT
1483	001650 045476	EM309
1484	001652 050442	EM4000
1485	001654 041074	DT003
1486	001656 041456	DF025
1487	:	ERROR 37: ATTEMPTING TO CHECK HEADER GENERATION
1488	:	WITH VARIOUS FORMAT VALUES
1489	:	FIRST WORD OF HEADER INCORRECT
1490	001660 045476	EM309
1491	001662 050522	EM4003
1492	001664 041074	DT003
1493	001666 041456	DF025
1494	:	ERROR 40: ATTEMPTING TO CHECK HEADER GENERATION
1495	:	WITH VARIOUS FORMAT VALUES
1496	:	SECOND WORD OF HEADER INCORRECT
1497	001670 045476	EM309
1498	001672 050552	EM4004
1499	001674 041074	DT003
1500	001676 041456	DF025
1501	:	ERROR 41: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA
1502	:	CS1 INCORRECT
1503	001700 045576	EM310
1504	001702 050442	EM4000
1505	001704 041114	DT041
1506	001706 041502	DF041
1507	:	ERROR 42: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA

1508	:	CS2 INCORRECT	
1509	001710	045576	EM310
1510	001712	050602	EM4005
1511	001714	041114	DT041
1512	001716	041502	DF041
1513	:	ERROR 43: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA	
1514	:	ERROR REGISTER INCORRECT	
1515	001720	045576	EM310
1516	001722	050620	EM4006
1517	001724	041114	DT041
1518	001726	041502	DF041
1519	:	ERROR 44: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA	
1520	:	BUS ADD INCORRECT	
1521	001730	045576	EM310
1522	001732	050644	EM4007
1523	001734	041114	DT041
1524	001736	041502	DF041
1525	:	ERROR 45: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA	
1526	:	WORD COUNT INCORRECT	
1527	001740	045576	EM310
1528	001742	050672	EM4008
1529	001744	041114	DT041
1530	001746	041502	DF041
1531	:	ERROR 46: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA	
1532	:	CS1 INCORRECT AFTER READING DATA BUFFER	
1533	001750	045576	EM310
1534	001752	050717	EM4009
1535	001754	041144	DT046
1536	001756	041536	DF046
1537	:	ERROR 47: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA	
1538	:	CS2 INCORRECT AFTER READING DATA BUFFER	
1539	001760	045576	EM310
1540	001762	050767	EM4010
1541	001764	041144	DT046
1542	001766	041536	DF046
1543	:	ERROR 50: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA	
1544	:	ERROR REG INCORRECT AFTER READING DATA BUFFER	
1545	001770	045576	EM310
1546	001772	051037	EM4011
1547	001774	041144	DT046
1548	001776	041536	DF046
1549	:	ERROR 51: ATTEMPTING TO CHECK NPR TRANSFER FOR WRITE DATA	
1550	:	DATA READ FROM MEMORY INCORRECT	
1551	002000	045576	EM310
1552	002002	051115	EM4012
1553	002004	041164	DT051
1554	002006	041562	DF051
1555	:	ERROR 52: ATTEMPTING TO CHECK HEADER RECOGNITION	
1556	:	CS1 INCORRECT	
1557	002010	045663	EM311
1558	002012	050442	EM4000
1559	002014	041176	DT052
1560	002016	041606	DF052
1561	:	ERROR 53: ATTEMPTING TO CHECK HEADER RECOGNITION	
1562	:	MR1 INCORRECT AFTER GAP IN WRITE DATA	
1563	002020	045663	EM311

1564	002022	051155	EM4013
1565	002024	041212	DT053
1566	002026	041632	DF053
1567	:	:	ERROR 54: ATTEMPTING TO CHECK SECTOR INCREMENT
1568	:	:	DISK ADDRESS REG INCORRECT
1569	002030	045732	EM312
1570	002032	051223	EM4014
1571	002034	041222	DT054
1572	002036	041656	DF054
1573	:	:	ERROR 55: ATTEMPTING TO CHECK SECTOR INCREMENT
1574	:	:	CYLINDER ADDRESS INCORRECT
1575	002040	045732	EM312
1576	002042	051257	EM4015
1577	002044	041222	DT054
1578	002046	041656	DF054
1579	:	:	ERROR 56: ATTEMPTING TO CHECK TRACK INCREMENT
1580	:	:	DISK ADDRESS REG INCORRECT
1581	002050	045777	EM313
1582	002052	051223	EM4014
1583	002054	041222	DT054
1584	002056	041656	DF054
1585	:	:	ERROR 57: ATTEMPTING TO CHECK TRACK INCREMENT
1586	:	:	CYLINDER ADDRESS INCORRECT
1587	002060	045777	EM313
1588	002062	051257	EM4015
1589	002064	041222	DT054
1590	002066	041656	DF054
1591	:	:	ERROR 60: ATTEMPTING TO CHECK CYLINDER INCREMENT
1592	:	:	DISK ADDRESS INCORRECT
1593	002070	046043	EM314
1594	002072	051223	EM4014
1595	002074	041222	DT054
1596	002076	041656	DF054
1597	:	:	ERROR 61: ATTEMPTING TO CHECK CYLINDER INCREMENT
1598	:	:	CYLINDER ADDRESS INCORRECT
1599	002100	046043	EM314
1600	002102	051257	EM4015
1601	002104	041222	DT054
1602	002106	041656	DF054
1603	:	:	ERROR 62: ATTEMPTING TO CHECK SECTOR PULSE DETECTION WITH
1604	:	:	WRITE DATA
1605	:	:	MAINT. REG. 1 INCORRECT
1606	002110	046112	EM315
1607	002112	051317	EM4016
1608	002114	041212	DT053
1609	002116	041632	DF053
1610	:	:	ERROR 63: ATTEMPTING TO FORCE BAD SECTOR ERROR
1611	:	:	CS1 INCORRECT
1612	002120	046206	EM316
1613	002122	050442	EM4000
1614	002124	041144	DT046
1615	002126	041536	DF046
1616	:	:	ERROR 64: ATTEMPTING TO FORCE BAD SECTOR ERROR
1617	:	:	CS2 INCORRECT
1618	002130	046206	EM316
1619	002132	050602	EM4005



1620	002134	041144	DT046
1621	002136	041536	DF046
1622	:	:	ERROR 65: ATTEMPTING TO FORCE BAD SECTOR ERROR
1623	:	:	ERROR REG INCORRECT
1624	002140	046206	EM316
1625	002142	050620	EM4006
1626	002144	041144	DT046
1627	002146	041536	DF046
1628	:	:	ERROR 66: ATTEMPTING TO FORCE HEADER VRC ERROR
1629	:	:	WHEN BAD SECTOR PRESENT
1630	:	:	CS1 INCORRECT
1631	002150	046253	EM317
1632	002152	050442	EM4000
1633	002154	041144	DT046
1634	002156	041536	DF046
1635	:	:	ERROR 67: ATTEMPTING TO FORCE HEADER VRC ERROR
1636	:	:	WHEN BAD SECTOR PRESENT
1637	:	:	CS2 INCORRECT
1638	002160	046253	EM317
1639	002162	050602	EM4005
1640	002164	041144	DT046
1641	002166	041536	DF046
1642	:	:	ERROR 70: ATTEMPTING TO FORCE HEADER VRC ERROR
1643	:	:	WHEN BAD SECTOR PRESENT
1644	:	:	ERROR REG INCORRECT
1645	002170	046253	EM317
1646	002172	050620	EM4006
1647	002174	041144	DT046
1648	002176	041536	DF046
1649	:	:	ERROR 71: ATTEMPTING TO FORCE HVRC ERROR
1650	:	:	CS1 INCORRECT
1651	002200	046351	EM318
1652	002202	050442	EM4000
1653	002204	041236	DT071
1654	002206	041702	DF071
1655	:	:	ERROR 72: ATTEMPTING TO FORCE HVRC ERROR
1656	:	:	CS2 INCORRECT
1657	002210	046351	EM318
1658	002212	050602	EM4005
1659	002214	041236	DT071
1660	002216	041702	DF071
1661	:	:	ERROR 73: ATTEMPTING TO FORCE HVRC ERROR
1662	:	:	ERROR REG INCORRECT
1663	002220	046351	EM318
1664	002222	050620	EM4006
1665	002224	041236	DT071
1666	002226	041702	DF071
1667	:	:	ERROR 74: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1668	:	:	CS1 INCORRECT
1669	002230	046410	EM319
1670	002232	050442	EM4000
1671	002234	041264	DT074
1672	002236	041736	DF074
1673	:	:	ERROR 75: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1674	:	:	CS2 INCORRECT
1675	002240	046410	EM319

1676	002242	050602	EM4005
1677	002244	041264	DT074
1678	002246	041736	DF074
1679	:	:	ERROR 76: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1680	:	:	ERROR REG INCORRECT
1681	002250	046410	EM319
1682	002252	050620	EM4006
1683	002254	041264	DT074
1684	002256	041736	DF074
1685	:	:	ERROR 77: ATTEMPTING TO FORCE OPERATION INCOMPLETE
1686	:	:	MR1 INCORRECT AFTER GAP IN WRITE DATA
1687	002260	046410	EM319
1688	002262	051155	EM4013
1689	002264	041306	DT077
1690	002266	041762	DF077
1691	:	:	ERROR 100: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1692	:	:	CS1 INCORRECT
1693	002270	046461	EM320
1694	002272	050442	EM4000
1695	002274	041264	DT074
1696	002276	041736	DF074
1697	:	:	ERROR 101: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1698	:	:	CS2 INCORRECT
1699	002300	046461	EM320
1700	002302	050602	EM4005
1701	002304	041264	DT074
1702	002306	041736	DF074
1703	:	:	ERROR 102: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1704	:	:	ERROR REG INCORRECT
1705	002310	046461	EM320
1706	002312	050620	EM4006
1707	002314	041264	DT074
1708	002316	041736	DF074
1709	:	:	ERROR 103: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 37
1710	:	:	MR1 INCORRECT AFTER GAP IN WRITE DATA
1711	002320	046461	EM320
1712	002322	051155	EM4013
1713	002324	041306	DT077
1714	002326	041762	DF077
1715	:	:	ERROR 104: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1716	:	:	CS1 INCORRECT
1717	002330	046545	EM321
1718	002332	050442	EM4000
1719	002334	041264	DT074
1720	002336	041736	DF074
1721	:	:	ERROR 105: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1722	:	:	CS2 INCORRECT
1723	002340	046545	EM321
1724	002342	050602	EM4005
1725	002344	041264	DT074
1726	002346	041736	DF074
1727	:	:	ERROR 106: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1728	:	:	ERROR REG INCORRECT
1729	002350	046545	EM321
1730	002352	050620	EM4006
1731	002354	041264	DT074

1732	002356	041736	DF074
1733			ERROR 107: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36
1734			MR1 INCORRECT AFTER GAP IN WRITE DATA
1735	002360	046545	EM321
1736	002362	051155	EM4013
1737	002364	041306	DT077
1738	002366	041762	DF077
1739			ERROR 110: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1740			CS1 INCORRECT
1741	002370	046631	EM322
1742	002372	050442	EM4000
1743	002374	041264	DT074
1744	002376	041736	DF074
1745			ERROR 111: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1746			CS2 INCORRECT
1747	002400	046631	EM322
1748	002402	050602	EM4005
1749	002404	041264	DT074
1750	002406	041736	DF074
1751			ERROR 112: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1752			ERROR REG INCORRECT
1753	002410	046631	EM322
1754	002412	050620	EM4006
1755	002414	041264	DT074
1756	002416	041736	DF074
1757			ERROR 113: ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0
1758			MR1 INCORRECT AFTER GAP IN WRITE DATA
1759	002420	046631	EM322
1760	002422	051155	EM4013
1761	002424	041306	DT077
1762	002426	041762	DF077
1763			ERROR 114: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1764			SECTOR ERROR CS1 INCORRECT
1765	002430	046714	EM323
1766	002432	050442	EM4000
1767	002434	041264	DT074
1768	002436	041736	DF074
1769			ERROR 115: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1770			SECTOR ERROR CS2 INCORRECT
1771	002440	046714	EM323
1772	002442	050602	EM4005
1773	002444	041264	DT074
1774	002446	041736	DF074
1775			ERROR 116: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1776			SECTOR ERROR ERROR INCORRECT
1777	002450	046714	EM323
1778	002452	050620	EM4006
1779	002454	041264	DT074
1780	002456	041736	DF074
1781			ERROR 117: CHECKING HEADER RECOGNITION WITH PREVIOUS BAD
1782			SECTOR ERROR MR1 INCORRECT
1783	002460	046714	EM323
1784	002462	051155	EM4013
1785	002464	041306	DT077
1786	002466	041762	DF077
1787			ERROR 120: CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER

1788			:		VRC ERROR CS1 INCORRECT
1789	002470	047007	:	EM324	
1790	002472	050442	:	EM4000	
1791	002474	041264	:	DT074	
1792	002476	041736	:	DF074	
1793			:	ERROR 121:	CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1794			:		VRC ERROR CS2 INCORRECT
1795	002500	047007	:	EM324	
1796	002502	050602	:	EM4005	
1797	002504	041264	:	DT074	
1798	002506	041736	:	DF074	
1799			:	ERROR 122:	CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1800			:		VRC ERROR ERROR REG INCORRECT
1801	002510	047007	:	EM324	
1802	002512	050620	:	EM4006	
1803	002514	041264	:	DT074	
1804	002516	041736	:	DF074	
1805			:	ERROR 123:	CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER
1806			:		VRC ERROR MR1 INCORRECT
1807	002520	047007	:	EM324	
1808	002522	051155	:	EM4013	
1809	002524	041306	:	DT077	
1810	002526	041762	:	DF077	
1811			:	ERROR 124:	FORCING BAD SECTOR ERROR WITH PREV HEADER
1812			:		VRC ERROR CS1 INCORRECT
1813	002530	047102	:	EM325	
1814	002532	050442	:	EM4000	
1815	002534	041264	:	DT074	
1816	002536	041736	:	DF074	
1817			:	ERROR 125:	FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1818			:		CS2 INCORRECT
1819	002540	047102	:	EM325	
1820	002542	050602	:	EM4005	
1821	002544	041264	:	DT074	
1822	002546	041736	:	DF074	
1823			:	ERROR 126:	FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1824			:		ERROR REG INCORRECT
1825	002550	047102	:	EM325	
1826	002552	050620	:	EM4006	
1827	002554	041264	:	DT074	
1828	002556	041736	:	DF074	
1829			:	ERROR 127:	FORCING BAD SECTOR ERROR WITH PREV HEADER VRC ERROR
1830			:		MR1 INCORRECT
1831	002560	047102	:	EM325	
1832	002562	051155	:	EM4013	
1833	002564	041306	:	DT077	
1834	002566	041762	:	DF077	
1835			:	ERROR 130:	FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1836			:		CS1 INCORRECT
1837	002570	047172	:	EM326	
1838	002572	050442	:	EM4000	
1839	002574	041264	:	DT074	
1840	002576	041736	:	DF074	
1841			:	ERROR 131:	FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1842			:		CS2 INCORRECT
1843	002600	047172	:	EM326	

1844	002602	050602	EM4005
1845	002604	041264	DT074
1846	002606	041736	DF074
1847	:	:	ERROR 132: FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1848	:	:	ERROR REG INCORRECT
1849	002610	047172	EM326
1850	002612	050620	EM4006
1851	002614	041264	DT074
1852	002616	041736	DF074
1853	:	:	ERROR 133: FORCING HEADER VRC ERROR WITH PREV BAD SECTOR ERROR
1854	:	:	MR1 INCORRECT
1855	002620	047172	EM326
1856	002622	051155	EM4013
1857	002624	041306	DT077
1858	002626	041762	DF077
1859	:	:	ERROR 134: ATTEMPTING TO CHECK ECC PATTERN CLEARING
1860	:	:	ECC PATTERN INCORRECT
1861	002630	047411	EM329
1862	002632	051335	EM4017
1863	002634	041320	DT134
1864	002636	042006	DF134
1865	:	:	ERROR 135: ATTEMPTING TO CHECK ECC GENERATION
1866	:	:	ECC PATTERN INCORRECT
1867	002640	047462	EM330
1868	002642	051335	EM4017
1869	002644	041320	DT134
1870	002646	042006	DF134
1871	:	:	ERROR 136: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1872	:	:	CS1 INCORRECT
1873	002650	047525	EM331
1874	002652	050442	EM4000
1875	002654	041144	DT046
1876	002656	041536	DF046
1877	:	:	ERROR 137: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1878	:	:	CS2 INCORRECT
1879	002660	047525	EM331
1880	002662	050602	EM4005
1881	002664	041144	DT046
1882	002666	041536	DF046
1883	:	:	ERROR 140: ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR
1884	:	:	ERROR REG. INCORRECT
1885	002670	047525	EM331
1886	002672	050620	EM4006
1887	002674	041144	DT046
1888	002676	041536	DF046
1889	:	:	ERROR 141: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1890	:	:	CS1 INCORRECT
1891	002700	047612	EM332
1892	002702	050442	EM4000
1893	002704	041144	DT046
1894	002706	041536	DF046
1895	:	:	ERROR 142: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1896	:	:	CS2 INCORRECT
1897	002710	047612	EM332
1898	002712	050602	EM4005
1899	002714	041144	DT046

1900	002716	041536	DF046
1901	:	:	ERROR 143: ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR
1902	:	:	ERROR REG. INCORRECT
1903	002720	047612	EM332
1904	002722	050620	EM4006
1905	002724	041144	DT046
1906	002726	041536	DF046
1907	:	:	ERROR 144: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1908	:	:	CS1 INCORRECT
1909	002730	047773	EM335
1910	002732	050442	EM4000
1911	002734	041332	DT144
1912	002736	042032	DF144
1913	:	:	ERROR 145: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1914	:	:	CS2 INCORRECT
1915	002740	047773	EM335
1916	002742	050602	EM4005
1917	002744	041332	DT144
1918	002746	042032	DF144
1919	:	:	ERROR 146: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1920	:	:	ERROR REG. INCORRECT
1921	002750	047773	EM335
1922	002752	050620	EM4006
1923	002754	041332	DT144
1924	002756	042032	DF144
1925	:	:	ERROR 147: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1926	:	:	BUS ADDRESS INCORRECT
1927	002760	047773	EM335
1928	002762	050644	EM4007
1929	002764	041332	DT144
1930	002766	042032	DF144
1931	:	:	ERROR 150: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1932	:	:	WORD COUNT INCORRECT
1933	002770	047773	EM335
1934	002772	050672	EM4008
1935	002774	041332	DT144
1936	002776	042032	DF144
1937	:	:	ERROR 151: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1938	:	:	CYLINDER ADD INCORRECT
1939	003000	047773	EM335
1940	003002	051257	EM4015
1941	003004	041332	DT144
1942	003006	042032	DF144
1943	:	:	ERROR 152: ATTEMPTING COMPLETE EXECUTION OF WRITE DATA
1944	:	:	DISK ADDRESS INCORRECT
1945	003010	047773	EM335
1946	003012	051223	EM4014
1947	003014	041332	DT144
1948	003016	042032	DF144
1949	:	:	ERROR 153: ATTEMPTING ERROR CLEAR WITH CONTROLLER CLEAR
1950	:	:	CS1 INCORRECT
1951	003020	050047	EM336
1952	003022	050442	EM4000
1953	003024	041176	DT052
1954	003026	041606	DF052
1955	:	:	ERROR 154: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL

1956			:	CS1 INCORRECT
1957	003030	050124		EM337
1958	003032	050442		EM4000
1959	003034	041332		DT144
1960	003036	042032		DF144
1961			:	ERROR 155: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
1962			:	CS2 INCORRECT
1963	003040	050124		EM337
1964	003042	050602		EM4005
1965	003044	041332		DT144
1966	003046	042032		DF144
1967			:	ERROR 156: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
1968			:	ERROR REGISTER INCORRECT
1969	003050	050124		EM337
1970	003052	050620		EM4006
1971	003054	041332		DT144
1972	003056	042032		DF144
1973			:	ERROR 157: ATTEMPTIN PARTIAL SECTOR WRITE WITH ZERO FILL
1974			:	BUS ADDRESS INCORRECT
1975	003060	050124		EM337
1976	003062	050644		EM4007
1977	003064	041332		DT144
1978	003066	042032		DF144
1979			:	ERROR 160: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
1980			:	WORD COUNT INCORRECT
1981	003070	050124		EM337
1982	003072	050672		EM4008
1983	003074	041332		DT144
1984	003076	042032		DF144
1985			:	ERROR 161: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
1986			:	CYLINDER ADDRESS INCORRECT
1987	003100	050124		EM337
1988	003102	051257		EM4015
1989	003104	041332		DT144
1990	003106	042032		DF144
1991			:	ERROR 162: ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL
1992			:	DISK ADDRESS INCORRECT
1993	003110	050124		EM337
1994	003112	051223		EM4014
1995	003114	041332		DT144
1996	003116	042032		DF144
1997			:	ERROR 163: ATTEMPTING WRITE DATA IN 18 BIT MODE
1998			:	RKECPT INCORRECT
1999	003120	050375		EM340
2000	003122	051335		EM4017
2001	003124	041372		DT151
2002	003126	042066		DF151
2003			:	ERROR 164: ATTEMPTING WRITE DATA IN 18 BIT MODE
2004			:	RKECPT INCORRECT
2005	003130	050375		EM340
2006	003132	051335		EM4017
2007	003134	041320		DT134
2008	003136	042006		DF134
2009				
2010				
2011				

.SBTTL TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER

2012	003140	000000	T.CS1:	.WORD	0	;CONTROL AND STATUS REGISTER 1
2013	003142	000000	T.WC:	.WORD	0	;WORD COUNT REGISTER
2014	003144	000000	T.BA:	.WORD	0	;BUS ADDRESS REGISTER
2015	003146	000000	T.DA:	.WORD	0	;DESIRED TRACK SECTOR REGISTER
2016	003150	000000	T.CS2:	.WORD	0	;CONTROL AND STATUS REGISTER 2
2017	003152	000000	T.DS:	.WORD	0	;DRIVE STATUS REGISTER
2018	003154	000000	T.ER:	.WORD	0	;ERROR REGISTER
2019	003156	000000	T.ASOF:	.WORD	0	;ATTENTION SUMMARY AND OFFSET REGISTER
2020	003160	000000	T.DCYL:	.WORD	0	;DESIRED CYLINDER REGISTER
2021	003162	000000	T.DB:	.WORD	0	;DATA BUFFER
2022	003164	000000	T.MR1:	.WORD	0	;MAINTENANCE REGISTER 1
2023	003166	000000	T.MR2:	.WORD	0	;MAINTENANCE REGISTER 2
2024	003170	000000	T.MR3:	.WORD	0	;MAINTENANCE REGISTER 3
2025	003172	000000	T.ECPS:	.WORD	0	;ECC POSITION INFORMATION
2026	003174	000000	T.ECPT:	.WORD	0	;ECC PATTERN INFORMATION
2027	003176	000000	T.SPARE:	.WORD	0	;SPARE REGISTER
2028						
2029			.SBTTL	EXPECTED RK611 CONTROLLER REGISTERS		
2030						
2031	003200	000000	E.CS1:	.WORD	0	;CONTROL AND STATUS REGISTER 1
2032	003202	000000	E.WC:	.WORD	0	;WORD COUNT REGISTER
2033	003204	000000	E.BA:	.WORD	0	;BUS ADDRESS REGISTER
2034	003206	000000	E.DA:	.WORD	0	;DESIRED TRACK SECTOR REGISTER
2035	003210	000000	E.CS2:	.WORD	0	;CONTROL AND STATUS REGISTER 2
2036	003212	000000	E.DS:	.WORD	0	;DRIVE STATUS REGISTER
2037	003214	000000	E.ER:	.WORD	0	;ERROR REGISTER
2038	003216	000000	E.ASOF:	.WORD	0	;ATTENTION SUMMARY AND OFFSET REGISTER
2039	003220	000000	E.DCYL:	.WORD	0	;DESIRED CYLINDER REGISTER
2040	003222	000000	E.DB:	.WORD	0	;DATA BUFFER
2041	003224	000000	E.MR1:	.WORD	0	;MAINTENANCE REGISTER 1
2042	003226	000000	E.MR2:	.WORD	0	;MAINTENANCE REGISTER 2
2043	003230	000000	E.MR3:	.WORD	0	;MAINTENANCE REGISTER 3
2044	003232	000000	E.ECPS:	.WORD	0	;ECC POSITION INFORMATION
2045	003234	000000	E.ECPT:	.WORD	0	;ECC PATTERN INFORMATION
2046	003236	000000	E.SPARE:	.WORD	0	;SPARE REGISTER
2047			.SBTTL	PROGRAM DEFINED VARIABLES		
2048						
2049	003240	000214	RKVEC:	.WORD	214	;RK611 VECTOR ADDRESS
2050	003242	000240	RKPRI:	.WORD	PR5	;RK611 PRIORITY
2051	003244	000000	TRAPPC:	.WORD	0	;TRAP PC FOR MEMORY PARITY OPTION
2052	003246	000000	SRTFLG:	.WORD	0	;START FLAG
2053						: 0 = 200
2054						: 1 = 214
2055						: -1 = 204
2056	003250	000000	ERRCNT:	.WORD	0	;ERROR COUNT FOR ABORT ERROR THRESHOLD
2057	003252	000000	P1.BIT:	.WORD	0	;NEXT BIT
2058	003254	000000	PR.BIT:	.WORD	0	;PRESENT BIT
2059	003256	000000	M1.BIT:	.WORD	0	;BIT-1
2060	003260	000000	M2.BIT:	.WORD	0	;BIT-2
2061	003262	000000	BITCNT:	.WORD	0	;BIT COUNT
2062	003264	000000	WRDCNT:	.WORD	0	;WORD COUNT FOR NPR TRANSFERS
2063	003266	000000	ECCHI:	.WORD	0	;16 MOST SIGNIFICANT BIT OF ECC
2064	003270	000000	ECCLO:	.WORD	0	;16 LEAST SIGNIFICANT BIT OF ECC
2065	003272	000000	ECCXOR:	.WORD	0	;FLAG FOR ECC GENERATION
2066	003274	000000	MEMPAR:	.WORD	0	;FLAG FOR MEMORY PARITY OPTION
2067	003276	000	SECTOR:	.BYTE	0	;SECTOR FOR INCREMENT TEST



```
2068 003277 000 TRACK: .BYTE 0 ;TRACK FOR INCREMENT TEST
2069 003300 000000 CYLN: .WORD 0 ;CYLINDER FOR INCREMENT TEST
2070 003302 000000 000000 000000 HEADER: .WORD 0,0,0 ;EXPECTED FOR INCREMENT TEST
2071 003310 000000 HDRCNT: .WORD 0 ;NUMBER OF HEADERS FOR OPI'S
2072 003312 000000 SAVSWR: .WORD 0 ;STORAGE FOR SWITCH REGISTER
2073 003314 000000 SEC24: .WORD 0 ;FLAG FOR 18-BIT WRITE DATA TESTS
2074 .SBTTL PROGRAM SETUP
2075
2076 003316 012737 000001 003246 PARM: MOV #1,SRTFLG ;LOAD START FLAG FOR PARAMETER START
2077 003324 000406 BR START1
2078
2079 003326 012737 177777 003246 P.RESTR: MOV #-1,SRTFLG ;LOAD START FLAG FOR RESTART
2080 003334 000402 BR START1
2081
2082 003336 005037 003246 START: CLR SRTFLG ;CLEAR START FLAG
2083 003342 000005 START1: RESET ;RESET THE WHOLE SYSTEM
2084 003344 105037 001103 CLRBR $ERFLG ;CLEAR ERROR FLAG
2085 003350 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
2086 003354 004737 037144 JSR PC,$TKINT ;INIT KEYBOARD
2087 003360 012746 000340 MOV #PR7,-(SP) ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
2088 003364 012746 003372 MOV #1$,-(SP) ;LOAD START OF PROGRAM
2089 003370 000002 RTI ;LOAD PSW
2090
2091 003372 1$:
2092 .SBTTL INITIALIZE THE COMMON TAGS
2093 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2094 003372 012706 001100 MOV #$CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
2095 003376 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
2096 003400 022706 001140 CMP #SWR,R6 ;;DONE?
2097 003404 001374 BNE -6 ;;LOOP BACK IF NO
2098 003406 012706 001100 MOV #STACK,SP ;;SETUP THE STACK POINTER
2099 ;;INITIALIZE A FEW VECTORS
2100 003412 012737 033246 000020 MOV #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
2101 003420 012737 000340 000022 MOV #340,@#IOTVEC+2 ;;LEVEL 7
2102 003426 012737 036000 000030 MOV #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
2103 003434 012737 000340 000032 MOV #340,@#EMTVEC+2 ;;LEVEL 7
2104 003442 012737 040756 000034 MOV #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2105 003450 012737 000340 000036 MOV #340,@#TRAPVEC+2;LEVEL 7
2106 003456 012737 040626 000024 MOV #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
2107 003464 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;LEVEL 7
2108 003472 013737 033014 033006 MOV $ENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
2109 003500 005037 001200 CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
2110 003504 005037 001202 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2111 003510 112737 000001 001115 MOVB #1,$ERMAX ;;ALLOW ONE ERROR PER TEST
2112 003516 012737 003516 001106 MOV #,$LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2113 003524 012737 003524 001110 MOV #,$LPERR ;;SETUP THE ERROR LOOP ADDRESS
2114
2115 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2116 003532 013746 000004 ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2117 003536 012737 003572 000004 MOV @#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
2118 003544 012737 177570 001140 MOV #64$,@#ERRVEC ;;SET UP ERROR VECTOR
2119 003552 012737 177570 001142 MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
2120 003560 022777 177777 175352 MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
2121 003566 001012 BNE 66$ ;;TRY TO REFERENCE HARDWARE SWR
2122 ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2123 003570 000403 BR 65$ ;;AND THE HARDWARE SWR IS NOT = -1
;;BRANCH IF NO TIMEOUT
```

```

2124 003572 012716 003600      64$:  MOV    #65$, (SP)      ;;SET UP FOR TRAP RETURN
2125 003576 000002                RTI
2126 003600 012737 000176 001140  65$:  MOV    #SWREG, SWR      ;;POINT TO SOFTWARE SWR
2127 003606 012737 000174 001142  MOV    #DISPREG, DISPLAY
2128 003614 012637 000004      66$:  MOV    (SP)+, @#ERRVEC  ;;RESTORE ERROR VECTOR
2129
2130 003620 005037 001222                CLR    $PASS           ;;CLEAR PASS COUNT
2131 003624 132737 000200 001235  BITB   #APTSIZE, $ENVM  ;;TEST USER SIZE UNDER APT
2132 003632 001403                BEQ    67$            ;;YES, USE NON-APT SWITCH
2133 003634 012737 001236 001140  MOV    #SSWREG, SWR    ;;NO, USE APT SWITCH REGISTER
2134 003642
2135 003642 005037 003250      67$:  CLR    ERRCNT
2136      .SBTTL  TYPE PROGRAM NAME
2137      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2138 003646 005227 177777      INC    #-1            ;;FIRST TIME?
2139 003652 001045                BNE    68$            ;;BRANCH IF NO
2140 003654 022737 033150 000042  CMP    #SENDAD, @#42  ;;ACT-11?
2141 003662 001441                BEQ    68$            ;;BRANCH IF YES
2142 003664 104401 003732      TYPE   ,69$          ;;TYPE ASCIZ STRING
2143      .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
2144 003670 005737 000042      TST    @#42          ;;ARE WE RUNNING UNDER XXDP/ACT?
2145 003674 001012                BNE    70$            ;;BRANCH IF YES
2146 003676 123727 001234 000001  CMPB   $ENV, #1       ;;ARE WE RUNNING UNDER APT?
2147 003704 001406                BEQ    70$            ;;BRANCH IF YES
2148 003706 023727 001140 000176  CMP    SWR, #SWREG    ;;SOFTWARE SWITCH REG SELECTED?
2149 003714 001005                BNE    71$            ;;BRANCH IF NO
2150 003716 104406                GTSWR                ;;GET SOFT-SWR SETTINGS
2151 003720 000403                BR     71$
2152 003722 112737 000001 001134  70$:  MOVB   #1, $AUTOB    ;;SET AUTO-MODE INDICATOR
2153 003730      71$:
2154 003730 000416      BR     68$          ;;GET OVER THE ASCIZ
2155      ;;69$: .ASCIZ <CRLF>/CZR6DCO RK611 DSKLS PRT4/<CRLF>
2156 003766      68$:
2157 003766 005227 177777      INC    #-1            ;TYPE RUN TIME MESSAGE FIRST PASS ONLY
2158 003772 001002                BNE    2$
2159 003774 104401 042261      TYPE   ,OPR006
2160 004000 022737 000001 003246  2$:  CMP    #1, SRTFLG    ;CHECK IF PARAMETER START
2161 004006 001117                BNE    15$           ;NO, CONTINUE SETUP
2162 004010 104401 042112      5$:  TYPE   ,OPR001      ;TYPE 'RK611 BUS ADDRESS ( ) ='
2163 004014 013746 001270      MOV    $BASE, -(SP)  ;;SAVE $BASE FOR TYPEOUT
2164 004020 104402                TYPOC                ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2165 004022 104401 042141      TYPE   ,OPR002
2166 004026 104412                RDOCT
2167 004030 012637 001160      MOV    (SP)+, $TMPO  ;GET VALUE
2168 004034 001407                BEQ    7$
2169 004036 022737 160000 001160  CMP    #160000, $TMPO ;CHECK IF <CR>
2170 004044 101361                BHI    5$            ;CHECK IF IN I/O PAGE
2171 004046 013737 001160 001270  MOV    $TMPO, $BASE  ;LOAD NEW BUS ADDRESS
2172 004054 104401 042147      7$:  TYPE   ,OPR003      ;TYPE 'RK611 VECTOR ADDRESS ( ) ='
2173 004060 013746 001264      MOV    $VECT1, -(SP) ;GET VECTOR ADDRESS FOR TYPE OUT
2174 004064 042716 160000      BIC    #160000, (SP) ;CLEAR PRIORITY BITS
2175 004070 104402                TYPOC
2176 004072 104401 042141      TYPE   ,OPR002
2177 004076 104412                RDOCT
2178 004100 012637 001160      MOV    (SP)+, $TMPO  ;GET VALUE
2179 004104 001407                BEQ    10$           ;CHECK IF <CR>
    
```

CZ  
CZ

```

2180 004106 022737 001000 001160      CMP      #1000,$TMPO      ;CHECK IF LEGAL
2181 004114 101757                    BLOS    7$
2182 004116 013737 001160 001264      MOV     $TMPO,$VECT1    ;LOAD NEW VECTOR ADDRESS
2183 004124 104401 042177 10$:      TYPE    ,OPR004        ;TYPE 'RK611 PRIORITY ( ) ='
2184 004130 005046                    CLR     -(SP)           ;MAKE ROOM ON THE STACK
2185 004132 113716 001265      MOV     $VECT1+1,(SP)   ;GET PRIORITY
2186 004136 006216                    ASR     (SP)           ;SHIF RIGH 5 BITS
2187 004140 006216                    ASR     (SP)
2188 004142 006216                    ASR     (SP)
2189 004144 006216                    ASR     (SP)
2190 004146 006216                    ASR     (SP)
2191 004150 104402                    TYPOC
2192 004152 104401 042141      TYPE    ,OPR002
2193 004156 104412                    RDOCT                    ;GET VALUE
2194 004160 012637 001160      MOV     (SP)+,$TMPO
2195 004164 001430                    BEQ     15$
2196 004166 022737 000007 001160      CMP     #7,$TMPO      ;CHECK FOR DEFAULT
2197 004174 103753                    BLO    10$
2198 004176 022737 000004 001160      CMP     #4,$TMPO
2199 004204 101347                    BHI    10$
2200 004206 006337 001160      ASL     $TMPO          ;SHIFT 5 BITS LEFT
2201 004212 006337 001160      ASL     $TMPO
2202 004216 006337 001160      ASL     $TMPO
2203 004222 006337 001160      ASL     $TMPO
2204 004226 006337 001160      ASL     $TMPO
2205 004232 042737 160000 001264      BIC     #160000,$VECT1 ;GET PRIORITY BITS
2206 004240 153737 001160 001265      BISB   $TMPO,$VECT1+1
2207 004246 013737 001264 003240 15$:      MOV     $VECT1,RKVEC
2208 004254 042737 160000 003240      BIC     #160000,RKVEC  ;GET VECTOR
2209 004262 113737 001265 003242      MOV     $VECT1+1,RKPRI ;GET PRIORITY
2210 004270 013702 001270      MOV     $BASE,R2      ;SET '611 BASE
2211 004274 005037 001202      CLR     $ESCAPE       ;CLEAR ESCAPE
2212 004300 012746 000000  NEWPAS: MOV     #PRO,-(SP)    ;LOCK OUT INTERRUPTS
2213 004304 012746 004312      MOV     #TST1,-(SP)
2214 004310 000002      RTI

```

.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.
: *
*****

```

```

2227 004312 000004                    TST1:  SCOPE
2228 004314 012737 000062 001200      MOV     #50,$TIMES    ;;DO 50. ITERATIONS
2229 004322 013702 001270      MOV     $BASE,R2      ;LOAD RK611 BASE
2230 004326 012762 100000 000000      MOV     #CCLR,RKCS1(R2) ;CLEAR RK611
2231 004334 012762 000040 000026      MOV     #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2232 004342 012762 177777 000002      MOV     #-1,RKWC(R2)  ;LOAD WORD COUNT
2233 004350 012762 053606 000004      MOV     #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2234 004356 012762 001777 000020      MOV     #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2235 004364 012762 003400 000006      MOV     #3400,RKDA(R2) ;LOAD TRACK

```

```

2236 004372 012762 000007 000010      MOV      #7,RKCS2(R2)      ;LOAD DRIVE NUM.
2237 004400 012762 012021 000000      MOV      #CDT!CFMT!RDATA,RKCS1(R2) ;ISSUE RDATA WITH CDT SET IN
2238                                     ; 24 SECTOR FORMAT
2239 004406 012700 000016                                     ;CLOCK IN DRIVE MESSAGE
2240 004412 012762 000440 000026 1$:    MOV      #DMD!MCLK,RKMR1(R2)
2241 004420 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2242 004426 005300      DEC      R0
2243 004430 001370      BNE      1$
2244 004432 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1
2245 004440 016237 000034 003166      MOV      RKMR2(R2),T.MR2 ;STORE MAINT REG. 2 (MESS A)
2246 004446 016237 000036 003170      MOV      RKMR3(R2),T.MR3 ;STORE MAINT REG. 3 (MESS B)
2247 004454 012737 012021 003200      MOV      #CDT!CFMT!RDATA,E.CS1 ;LOAD EXPECTED CS1
2248 004462 012737 071027 003226      MOV      #S.SEEK!S.FMT!70007,E.MR2 ;LOAD EXPECTED MR2
2249 004470 012737 037760 003230      MOV      #37760,E.MR3 ;LOAD EXPECTED MR3
2250 004476 023737 003200 003140      CMP      E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2251 004504 001405      BEQ      2$ ;YES, CHECK MESSAGE A
2252 004506 104003      ERROR   3 ;CS1 INCORRECT
2253 004510 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2254 004516 000412      BR       TST2 ;GO TO NEXT TEST
2255
2256 004520 023737 003226 003166 2$:    CMP      E.MR2,T.MR2 ;CHECK MESS A CORRECT
2257 004526 001401      BEQ      3$ ;YES, CHECK MESSAGE B
2258 004530 104004      ERROR   4 ;MESS A INCORRECT
2259 004532 023737 003230 003170 3$:    CMP      E.MR3,T.MR3 ;CHECK MESS B CORRECT
2260 004540 001401      BEQ      TST2 ;YES, GO ON TO NEXT TEST
2261 004542 104005      ERROR   5 ;MESS B INCORRECT
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272

```

```

*****
: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.
:
:
*****

```

```

2273 004544 000004 TST2:  SCOPE
2274 004546 012737 000062 001200      MOV      #50,$TIMES ;DO 50. ITERATIONS
2275 004554 013702 001270      MOV      $BASE,R2 ;LOAD RK611 BASE
2276 004560 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2277 004566 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2278 004574 012762 177777 000002      MOV      #-1,RKWC(R2) ;LOAD WORD COUNT
2279 004602 012762 053606 000004      MOV      #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2280 004610 012762 001777 000020      MOV      #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2281 004616 012762 003400 000006      MOV      #3400,RKDA(R2) ;LOAD TRACK
2282 004624 012762 000007 000010      MOV      #7,RKCS2(R2) ;LOAD DRIVE NUM.
2283 004632 012762 012023 000000      MOV      #CDT!CFMT!WRDATA,RKCS1(R2) ;ISSUE WRDATA WITH CDT SET IN
2284                                     ; 24 SECTOR FORMAT
2285 004640 012700 000016                                     ;CLOCK IN DRIVE MESSAGE
2286 004644 012762 000440 000026 1$:    MOV      #DMD!MCLK,RKMR1(R2)
2287 004652 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2288 004660 005300      DEC      R0
2289 004662 001370      BNE      1$
2290 004664 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1
2291 004672 016237 000034 003166      MOV      RKMR2(R2),T.MR2 ;STORE MAINT REG. 2 (MESS A)

```

```
2292 004700 016237 000036 003170 MOV RKMR3(R2),T.MR3 ;STORE MAINT REG. 3 (MESS B)
2293 004706 012737 012023 003200 MOV #CDT!CFMT!WRDATA,E.CS1 ;LOAD EXPECTED CS1
2294 004714 012737 071227 003226 MOV #S.SEEK!S.RTC!S.FMT!70007,E.MR2 ;LOAD EXPECTED MR2
2295 004722 012737 037760 003230 MOV #37760,E.MR3 ;LOAD EXPECTED MR3
2296 004730 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2297 004736 001405 BEQ 2$ ;YES, CHECK MESSAGE A
2298 004740 104006 ERROR 6 ;CS1 INCORRECT
2299 004742 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2300 004750 000412 BR TST3 ;;GO TO NEXT TEST
2301
2302 004752 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2303 004760 001401 BEQ 3$ ;YES, CHECK MESSAGE B
2304 004762 104007 ERROR 7 ;MESS A INCORRECT
2305 004764 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2306 004772 001401 BEQ TST3 ;;YES, GO ON TO NEXT TEST
2307 004774 104010 ERROR 10 ;MESS B INCORRECT
2308
2309
```

```
*****
*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.
*****
```

```
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320 004776 000004 TST3: SCOPE
2321 005000 012737 000050 001200 MOV #50,$TIMES ;;DO 50 ITERATIONS
2322 005006 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2323 005012 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2324 005020 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2325 005026 012762 177777 000002 MOV #-1,RKWC(R2) ;LOAD WORD COUNT
2326 005034 012762 053606 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2327 005042 012762 001777 000020 MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2328 005050 012762 003400 000006 MOV #3400,RKDA(R2) ;LOAD TRACK
2329 005056 012762 000007 000010 MOV #7,RKCS2(R2) ;LOAD DRIVE NUM.
2330 005064 012762 012031 000000 MOV #CDT!CFMT!WRTCHK,RKCS1(R2) ;ISSUE WRTCHK WITH CDT SET IN
2331 ; 24 SECTOR FORMAT
2332 005072 012700 000016 MOV #3*4+2,R0 ;CLOCK IN DRIVE MESSAGE
2333 005076 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
2334 005104 012762 000040 000026 MOV #DMD,RKMR1(R2)
2335 005112 005300 DEC R0
2336 005114 001370 BNE 1$
2337 005116 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1
2338 005124 016237 000034 003166 MOV RKMR2(R2),T.MR2 ;STORE MAINT REG. 2 (MESS A)
2339 005132 016237 000036 003170 MOV RKMR3(R2),T.MR3 ;STORE MAINT REG. 3 (MESS B)
2340 005140 012737 012031 003200 MOV #CDT!CFMT!WRTCHK,E.CS1 ;LOAD EXPECTED CS1
2341 005146 012737 071027 003226 MOV #S.SEEK!S.FMT!70007,E.MR2 ;LOAD EXPECTED MR2
2342 005154 012737 037760 003230 MOV #37760,E.MR3 ;LOAD EXPECTED MR3
2343 005162 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2344 005170 001405 BEQ 2$ ;YES, CHECK MESSAGE A
2345 005172 104011 ERROR 11 ;CS1 INCORRECT
2346 005174 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2347 005202 000412 BR TST4 ;;GO TO NEXT TEST
```

```
2348  
2349 005204 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT  
2350 005212 001401 BEQ 3$ ;YES, CHECK MESSAGE B  
2351 005214 104012 ERROR 12 ;MESS A INCORRECT  
2352 005216 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT  
2353 005224 001401 BEQ TST4 ;YES, GO ON TO NEXT TEST  
2354 005226 104013 ERROR 13 ;MESS B INCORRECT  
2355  
2356  
2357 :*****  
2358 :*TEST 4 READ DATA CLEAR MESSAGE  
2359 :*  
2360 :* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT THE  
2361 :* CONTROLLER ON DIAGNOSTIC MODE. ISSUE A READ DATA TO AN  
2362 :* RK06, IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE  
2363 :* 7, WORD COUNT 177777. CLOCK IN THE SEEK MESSAGE AND THE  
2364 :* CLEAR MESSAGE AND MAKE SURE THE CLEAR MESSAGE IS CORRECT.  
2365 :*****  
2366 005230 000004 TST4: SCOPE  
2367 005232 012737 000062 001200 MOV #50,$TIMES ;;DO 50. ITERATIONS  
2368 005240 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE  
2369 005244 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611  
2370 005252 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE  
2371 005260 012762 177777 000002 MOV #-1,RKWC(R2) ;LOAD WORD COUNT  
2372 005266 012762 053606 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS  
2373 005274 012762 001777 000020 MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.  
2374 005302 012762 003400 000006 MOV #3400,RKDA(R2) ;LOAD TRACK  
2375 005310 012762 000007 000010 MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER  
2376 005316 012762 012021 000000 MOV #CDT!CFMT!RDDATA,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN  
2377 : ; 24 SECTOR FORMAT  
2378 005324 012700 000156 MOV #27.*4+2,R0 ;LOAD COUNT TO LOAD DRIVE CLEAR  
2379 005330 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)  
2380 005336 012762 000040 000026 MOV #DMD,RKMR1(R2)  
2381 005344 005300 DEC R0  
2382 005346 001370 BNE 1$  
2383 005350 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1  
2384 005356 016237 000034 003166 MOV RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)  
2385 005364 016237 000036 003170 MOV RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)  
2386 005372 012737 012021 003200 MOV #CDT!CFMT!RDDATA,E.CS1 ;LOAD EXPECTED CS1  
2387 005400 012737 071407 003226 MOV #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2  
2388 005406 005037 003230 CLR E.MR3 ;LOAD EXPECTED MAINT REG.  
2389 005412 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT  
2390 005420 001405 BEQ 2$ ;YES, CHECK MESSB  
2391 005422 104014 ERROR 14 ;CS1 INCORRECT  
2392 005424 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611  
2393 005432 000412 BR TST5 ;GO TO NEXT TEST  
2394  
2395 005434 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT  
2396 005442 001401 BEQ 3$ ;YES, CHECK MESS B  
2397 005444 104015 ERROR 15 ;MESS A INCORRECT  
2398 005446 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT  
2399 005454 001401 BEQ TST5 ;YES, GO ON TO NEXT TEST  
2400 005456 104016 ERROR 16 ;MESS B INCORRECT
```

```
2401  
2402 :*****  
2403 :*TEST 5 WRITE DATA AND DRIVE CLEAR MESSAGE
```

```

2404
2405
2406
2407
2408
2409
2410
2411
2412 005460 000004
2413 005462 012737 000062 001200
2414 005470 013702 001270
2415 005474 012762 100000 000000
2416 005502 012762 000040 000026
2417 005510 012762 177777 000002
2418 005516 012762 053606 000004
2419 005524 012762 001777 000020
2420 005532 012762 003400 000006
2421 005540 012762 000007 000010
2422 005546 012762 012023 000000
2423
2424 005554 012700 000156
2425 005560 012762 000440 000026 1$:
2426 005566 012762 000040 000026
2427 005574 005300
2428 005576 001370
2429 005600 016237 000000 003140
2430 005606 016237 000034 003166
2431 005614 016237 000036 003170
2432 005622 012737 012023 003200
2433 005630 012737 071407 003226
2434 005636 005037 003230
2435 005642 023737 003200 003140
2436 005650 001405
2437 005652 104017
2438 005654 012762 100000 000000
2439 005662 000412
2440
2441 005664 023737 003226 003166 2$:
2442 005672 001401
2443 005674 104020
2444 005676 023737 003230 003170 3$:
2445 005704 001401
2446 005706 104021
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459 005710 000004
  
```

```

*****
: * 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.
: *
: *
: *
: *****
TST5: SCOPE
MOV #50,$TIMES ;;DO 50. ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #-1,RKWC(R2) ;LOAD WORD COUNT
MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
MOV #3400,RKDA(R2) ;LOAD TRACK
MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER
MOV #CDT!CFMT!WRDATA,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
; 24 SECTOR FORMAT
MOV #27.*4+2,R0 ;LOAD COUNT TO LOAD DRIVE CLEAR
1$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND 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.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
CLR E.MR3 ;LOAD EXPECTED MAINT REG.
CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
BEQ 2$ ;YES, CHECK MESSB
ERROR 17 ;CS1 INCORRECT
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
BR TST6 ;;GO TO NEXT TEST
2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
BEQ 3$ ;YES, CHECK MESS B
ERROR 20 ;MESS A INCORRECT
3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
BEQ TST6 ;YES, GO ON TO NEXT TEST
ERROR 21 ;MESS B INCORRECT
  
```

```

*****
: *TEST 6 DRIVE CLEAR 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
: * THROUGH SEEK MESSAGE AND CLOCK IN DRIVE CLEAR.
: * MAKE SURE CORRECT MESSAGE IS ASSEMBLED.
: *
: *
: *
: *****
TST6: SCOPE
  
```

```

2460 005712 012737 000062 001200 MOV #50.,$TIMES ;;DO 50. ITERATIONS
2461 005720 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2462 005724 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2463 005732 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2464 005740 012762 177777 000002 MOV #-1,RKWC(R2) ;LOAD WORD COUNT
2465 005746 012762 053606 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2466 005754 012762 001777 000020 MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2467 005762 012762 003400 000006 MOV #3400,RKDA(R2) ;LOAD TRACK
2468 005770 012762 000007 000010 MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER
2469 005776 012762 012031 000000 MOV #CDT!CFMT!WRTCHK,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2470 ; 24 SECTOR FORMAT
2471 006004 012700 000156 MOV #27.*4+2,R0 ;LOAD COUNT TO LOAD DRIVE CLEAR
2472 006010 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
2473 006016 012762 000040 000026 MOV #DMD,RKMR1(R2)
2474 006024 005300 DEC R0
2475 006026 001370 BNE 1$
2476 006030 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2477 006036 016237 000034 003166 MOV RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2478 006044 016237 000036 003170 MOV RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2479 006052 012737 012031 003200 MOV #CDT!CFMT!WRTCHK,E.CS1 ;LOAD EXPECTED CS1
2480 006060 012737 071407 003226 MOV #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2481 006066 005037 003230 CLR E.MR3 ;LOAD EXPECTED MAINT REG.
2482 006072 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2483 006100 001405 BEQ 2$ ;YES, CHECK MESSB
2484 006102 104022 ERROR 22 ;CS1 INCORRECT
2485 006104 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2486 006112 000412 BR TST7 ;GO TO NEXT TEST
2487
2488 006114 023737 003226 003166 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2489 006122 001401 BEQ 3$ ;YES, CHECK MESS B
2490 006124 104023 ERROR 23 ;MESS A INCORRECT
2491 006126 023737 003230 003170 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2492 006134 001401 BEQ TST7 ;YES, GO ON TO NEXT TEST
2493 006136 104024 ERROR 24 ;MESS B INCORRECT
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508

```

.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.
*****

```

```

2509 006140 000004 TST7: SCOPE
2510 006142 012737 000012 001200 MOV #10.,$TIMES ;;DO 10. ITERATIONS
2511 006150 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2512 006154 005037 003220 CLR E.DCYL ;INITIALIZE CYLINDER ADD
2513 006160 005037 003206 CLR E.DA ;INITIALIZE TRACK AND SECTOR
2514 006164 012737 000021 003200 MOV #RDDATA,E.CS1 ;INITIALIZE FORMAT
2515 006172 012737 006200 001110 MOV #1$, $LPERR ;LOAD LOOP ON ERROR LOCATION FOR

```



: SUBTEST LOOP

```
2516  
2517  
2518 006200  
2519 006200 004737 034402 1$: JSR PC,CALHDR ;CALULATE EXPECTED HEADER  
2520 006204 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611  
2521 006212 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IM MAINTENANCE MODE  
2522 006220 012762 177777 000002 MOV #-1,RKWC(R2) ;LOAD WORD COUNT  
2523 006226 012762 053606 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS  
2524 006234 013762 003220 000020 MOV E.DCYL,RKDCYL(R2) ;LOAD CYLINDER NUMBER  
2525 006242 013762 003206 000006 MOV E.DA,RKDA(R2) ;LOAD TRACK AND SECTOR  
2526 006250 013762 003200 000000 MOV E.CS1,RKCS1(R2) ;LOAD COMMAND AND FORMAT  
2527 006256 012700 000426 MOV #69.*4+2,R0 ;LOAD COUNT UNTIL HEADER GENERATION  
2528 006262 012762 000440 000026 5$: MOV #DMD!MCLK,RKMR1(R2)  
2529 006270 012762 000040 000026 MOV #DMD,RKMR1(R2)  
2530 006276 005300 DEC R0  
2531 006300 001370 BNE 5$  
2532 006302 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1  
2533 006310 016237 000034 003166 MOV RKMR2(R2),T.MR2 ;STORE FIRST HEADER WORD  
2534 006316 016237 000036 003170 MOV RKMR3(R2),T.MR3 ;STORE SECOND WORD  
2535 006324 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT  
2536 006332 001405 BEQ 6$ ;YES, CHECK FIRST LINE OF HEADER  
2537 006334 104025 ERROR 25 ;CS1 INCORRECT  
2538 006336 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER  
2539 006344 000412 BR 15$ ;YES, CHECK SECOND WORD OF HEADER  
2540  
2541 006346 023737 003226 003166 6$: CMP E.MR2,T.MR2 ;CHECK IF FIRST WORD OF HEADER CORRECT  
2542 006354 001401 BEQ 7$ ;YES, CHECK IF SECOND WORD OF HEADER CORRECT  
2543 006356 104026 ERROR 26 ;FIRST WORD OF HEADER INCORRECT  
2544 006360 023737 003230 003170 7$: CMP E.MR3,T.MR3 ;CHECK IF SECOND WORD OF HEADER CORRECT  
2545 006366 001401 BEQ 15$ ;YES,CHECK IF LOOP ON ERROR  
2546 006370 104027 ERROR 27 ;SECOND WORD INCORRECT  
2547 006372 104415 15$: SCOPE1 ;LOOP ON ERROR  
2548 006374 005237 003220 INC E.DCYL ;INCREMENT EXPECTED CYLINDER  
2549 006400 022737 000633 003220 CMP #633,E.DCYL ;CHECK IF VALUE OF CYLINDER OVERFLOW DETECTION  
2550 006406 001002 BNE 16$ ;NO, CONTINUE  
2551 006410 005237 003220 INC E.DCYL ;USE 634  
2552 006414 022737 001777 003220 16$: CMP #1777,E.DCYL ;CHECK IF FINISHED  
2553 006422 103266 BHIS 1$ ;NO, GO TO NEXT CONFIGURATION  
2554  
2555  
2556  
2557  
2558  
2559  
2560  
2561  
2562  
2563  
2564  
2565
```

```
*****  
: *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 1-7.  
: *
```

```
*****
```

```
2566 006424 000004 1$T10: SCOPE  
2567 006426 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS  
2568 006434 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE  
2569 006440 005037 003220 CLR E.DCYL ;INITIALIZE CYLINDER ADD  
2570 006444 005037 003206 CLR E.DA ;INITIALIZE TRACK AND SECTOR  
2571 006450 012737 000021 003200 MOV #RDATA,E.CS1 ;INITIALIZE FORMAT
```



```

2628
2629 006734          18:
2630 006734 004737 034402      JSR   PC,CALHDR      ;CALULATE EXPECTED HEADER
2631 006740 012762 100000 000000  MOV   #CCLR,RKCS1(R2) ;CLEAR RK611
2632 006746 012762 000040 000026  MOV   #DMD,RKMR1(R2) ;PUT RK611 IM MAINTENANCE MODE
2633 006754 012762 177777 000002  MOV   #-1,RKWC(R2)   ;LOAD WORD COUNT
2634 006762 012762 053606 000004  MOV   #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2635 006770 013762 003220 000020  MOV   E.DCYL,RKDCYL(R2) ;LOAD CYLINDER NUMBER
2636 006776 013762 003206 000006  MOV   E.DA,RKDA(R2)   ;LOAD TRACK AND SECTOR
2637 007004 013762 003200 000000  MOV   E.CS1,RKCS1(R2) ;LOAD COMMAND AND FORMAT
2638 007012 012700 000426          MOV   #69.*4+2,R0    ;LOAD COUNT UNTIL HEADER GENERATION
2639 007016 012762 000440 000026 58:   MOV   #DMD!MCLK,RKMR1(R2)
2640 007024 012762 000040 000026  MOV   #DMD,RKMR1(R2)
2641 007032 005300          DEC   R0
2642 007034 001370          BNE   58
2643 007036 016237 000000 003140  MOV   RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
2644 007044 016237 000034 003166  MOV   RKMR2(R2),T.MR2 ;STORE FIRST HEADER WORD
2645 007052 016237 000036 003170  MOV   RKMR3(R2),T.MR3 ;STORE SECOND WORD
2646 007060 023737 003200 003140  CMP   E.CS1,T.CS1    ;CHECK COMMAND AND STATUS REG 1 CORRECT
2647 007066 001405          BEQ   68              ;YES, CHECK FIRST LINE OF HEADER
2648 007070 104033          ERROR 33              ;CS1 INCORRECT
2649 007072 012762 100000 000000  MOV   #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2650 007100 000412          BR    158             ;YES, CHECK SECOND WORD OF HEADER
2651
2652 007102 023737 003226 003166 68:   CMP   E.MR2,T.MR2    ;CHECK IF FIRST WORD OF HEADER CORRECT
2653 007110 001401          BEQ   78              ;YES, CHECK IF SECOND WORD OF HEADER CORRECT
2654 007112 104034          ERROR 34              ;FIRST WORD OF HEADER INCORRECT
2655 007114 023737 003230 003170 78:   CMP   E.MR3,T.MR3    ;CHECK IF SECOND WORD OF HEADER CORRECT
2656 007122 001401          BEQ   158             ;YES,CHECK IF LOOP ON ERROR
2657 007124 104035          ERROR 35              ;SECOND WORD INCORRECT
2658 007126 104415          158: SCOPE1           ;LOOP ON ERROR
2659 007130 105237 003206          INCB  E.DA            ;INCREMENT SECTOR
2660 007134 122737 000025 003206  CMPB  #25,E.DA        ;CHECK IF FINISHED
2661 007142 103274          BHIS  18              ;NO, GO TO NEXT CONFIGURATION
2662
2663
2664 .....
2665 *TEST 12      HEADER GENERATION (PART 4)
2666 .....
2667 *
2668 *      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2669 *      THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA OF
2670 *      ONE WORD FOR AND RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
2671 *      HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR
2672 *      MESSAGES, CHECK MAINTENANCE REGISTER 2 AND MAINTENANCE
2673 *      REGISTER 3 TO MAKE SURE HEADER IS CORRECT. REPEAT FOR 24
2674 *      SECTOR FORMAT.
2675 .....
2676 TST12: SCOPE
2677 MOV   #50,$TIMES      ;;DO 50. ITERATIONS
2678 MOV   $BASE,R2        ;LOAD RK611 BASE
2679 CLR   E.DCYL          ;INITIALIZE CYLINDER ADD
2680 CLR   E.DA            ;INITIALIZE TRACK AND SECTOR
2681 MOV   #RDATA,E.CS1    ;INITIALIZE FORMAT
2682 MOV   #16,$LPERR      ;LOAD LOOP ON ERROR LOCATION FOR
2683 ; SUBTEST LOOP

```

```
2684 007204 004737 034402 18: JSR PC,CALHDR ;CALULATE EXPECTED HEADER
2685 007204 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2686 007210 012762 100000 000000 MOV #DMD,RKMR1(R2) ;PUT RK611 IM MAINTENANCE MODE
2687 007216 012762 000040 000026 MOV #-1,RKWC(R2) ;LOAD WORD COUNT
2688 007224 012762 177777 000002 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2689 007232 012762 053606 000004 MOV E.DCYL,RKDCYL(R2) ;LOAD CYLINDER NUMBER
2690 007240 013762 003220 000020 MOV E.DA,RKDA(R2) ;LOAD TRACK AND SECTOR
2691 007246 013762 003206 000006 MOV E.CS1,RKCS1(R2) ;LOAD COMMAND AND FORMAT
2692 007254 013762 003200 000000 MOV #69.*4+2,R0 ;LOAD COUNT UNTIL HEADER GENERATION
2693 007262 012700 000426 5$: MOV #DMD!MCLK,RKMR1(R2)
2694 007266 012762 000440 000026 MOV #DMD,RKMR1(R2)
2695 007274 012762 000040 000026 DEC R0
2696 007302 005300 BNE 5$
2697 007304 001370 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
2698 007306 016237 000000 003140 MOV RKMR2(R2),T.MR2 ;STORE FIRST HEADER WORD
2699 007314 016237 000034 003166 MOV RKMR3(R2),T.MR3 ;STORE SECOND WORD
2700 007322 016237 000036 003170 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
2701 007330 023737 003200 003140 BEQ 6$ ;YES, CHECK FIRST LINE OF HEADER
2702 007336 001405 ERROR 36 ;CS1 INCORRECT
2703 007340 104036 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2704 007342 012762 100000 000000 BR 15$ ;YES, CHECK SECOND WORD OF HEADER
2705 007350 000412
2706
2707 007352 023737 003226 003166 6$: CMP E.MR2,T.MR2 ;CHECK IF FIRST WORD OF HEADER CORRECT
2708 007360 001401 BEQ 7$ ;YES, CHECK IF SECOND WORD OF HEADER CORRECT
2709 007362 104037 ERROR 37 ;FIRST WORD OF HEADER INCORRECT
2710 007364 023737 003230 003170 7$: CMP E.MR3,T.MR3 ;CHECK IF SECOND WORD OF HEADER CORRECT
2711 007372 001401 BEQ 15$ ;YES,CHECK IF LOOP ON ERROR
2712 007374 104040 ERROR 40 ;SECOND WORD INCORRECT
2713 007376 104415 15$: SCOP1 ;LOOP ON ERROR
2714 007400 032737 010000 003200 BIT #CFMT,E.CS1 ;CHECK IF FINISHED
2715 007406 001004 BNE TST13 ;:YES, GO TO NEXT TEST
2716 007410 052737 010000 003200 BIS #CFMT,E.CS1 ;USE 24 SECTOR FORMAT
2717 007416 000672 BR 1$
2718
2719 .SBTTL **NPR TRANSFER FOR WRITE DATA
2720
2721 :*****
2722 :*TEST 13 WRITE DATA NPR TRANSFER
2723 :*
2724 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
2725 :* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
2726 :* 67 WORDS, TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
2727 :* TRACK 0, DRIVE 0. CLOCK IN SEEK AND DRIVE CLEAR MESSAGES.
2728 :* GIVE ENOUGH CLOCK PULSE FOR 68 SILO WORDS. MAKE SURE DATA
2729 :* LATE DOES NOT OCCUR. READ BACK 66 WORDS AND VERIFY THEY
2730 :* ARE CORRECT.
2731 :*
2732 :*****
2733 007420 000004 TST13: SCOPE
2734 007422 012737 000062 001200 MOV #50,$TIMES ;:DO 50. ITERATIONS
2735 007430 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2736 007434 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2737 007442 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2738 007450 012762 177675 000002 MOV #-67.,RKWC(R2) ;WORD COUNT=67
2739 007456 012762 051544 000004 MOV #NPRBUF,RKBA(R2) ;LOAD BUFFER ADDRESS
```

```

2740 007464 012762 000023 000000      MOV      #WRDATA,RKCS1(R2) ;ISSUE WRDATA
2741 007472 012700 004724          MOV      #68.*37.,R0      ;ISSUE ENOUGH CLOCKS FOR 68
2742 007476 012762 000440 000026 1$:    MOV      #DMD!MCLK,RKMR1(R2) ; NPR TRANSFERS
2743 007504 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2744 007512 005300          DEC      R0
2745 007514 001370          BNE     1$
2746 007516 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2747 007524 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
2748 007532 016237 000002 003142      MOV      RKWC(R2),T.WC   ;STORE WORD COUNT REG.
2749 007540 016237 000004 003144      MOV      RKBA(R2),T.BA   ;STORE BUS ADDRESS
2750 007546 016237 000014 003154      MOV      RKER(R2),T.ER   ;STORE ERROR REG.
2751 007554 012737 000023 003200      MOV      #WRDATA,E.CS1  ;LOAD EXPECTED CS1
2752 007562 012737 000200 003210      MOV      #OR,E.CS2      ;LOAD EXPECTED CS2
2753 007570 012737 177777 003202      MOV      #-1,E.WC       ;LOAD EXPECTED WORD COUNT
2754 007576 012737 051750 003204      MOV      #NPRBUF+<66.*2>,E.BA ;LOAD EXPECTED BUS ADDRESS
2755 007604 005037 003214          CLR      E.ER           ;LOAD EXPECTED ERROR REG
2756 007610 023737 003200 003140      CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
2757 007616 001405          BEQ     5$              ;YES, CHECK CS2
2758 007620 104041          ERROR   41              ;CS1 INCORRECT
2759 007622 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2760 007630 000517          BR      TST14          ;;GO TO NEXT TEST
2761
2762 007632 023737 003210 003150 5$:    CMP      E.CS2,T.CS2    ;CHECK CS2
2763 007640 001401          BEQ     6$              ;NO, CHECK BUS ADDRESS
2764 007642 104042          ERROR   42              ;CS2 INCORRECT
2765 007644 023737 003204 003144 6$:    CMP      E.BA,T.BA      ;CHECK BUS ADDRESS
2766 007652 001401          BEQ     7$              ;YES, CHECK WORD COUNT
2767 007654 104044          ERROR   44              ;BUS ADDRESS INCORRECT
2768 007656 023737 003202 003142 7$:    CMP      E.WC,T.WC      ;CHECK WORD COUNT
2769 007664 001401          BEQ     8$              ;YES, CONTINUE
2770 007666 104045          ERROR   45              ;WORD COUNT INCORRECT
2771 007670 023737 003214 003154 8$:    CMP      E.ER,T.ER      ;CHECK ERROR REG CORRECT
2772 007676 001401          BEQ     10$             ;YES, CONTINUE
2773 007700 104043          ERROR   43              ;ERROR REG INCORRECT
2774 007702 012703 051544          10$:    MOV      #NPRBUF,R3     ;LOAD ADDRESS OR START OF BUFFER
2775 007706 012701 000101          MOV      #65.,R1        ;LOAD COUNT FOR SICO
2776 007712 005037 003264          CLR      WRDCNT         ;INITIALIZE WORD COUNT
2777 007716 012737 000300 003210      MOV      #IR!OR,E.CS2   ;LOAD EXPECTED CS2
2778 007724 012337 003222          15$:    MOV      (R3)+,E.DB     ;LOAD EXPECTED DATA
2779 007730 016237 000024 003162      MOV      RKDB(R2),T.DB  ;GET DATA READ
2780 007736 012700 000020          MOV      #20,R0        ;SET COUNT TO WAIT FOR SILO
2781 007742 005300          16$:    DEC      R0            ;DEC COUNT
2782 007744 001376          BNE     16$            ;WAIT FOR 0
2783 007746 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;STORE CS1
2784 007754 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;STORE CS2
2785 007762 016237 000014 003154      MOV      RKER(R2),T.ER   ;STORE ERROR REG.
2786 007770 023737 003200 003140      CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
2787 007776 001405          BEQ     20$            ;YES, CONTINUE
2788 010000 104046          ERROR   46              ;CS1 INCORRECT
2789 010002 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2790 010010 000427          BR      TST14          ;;GO ON TO NEXT TEST
2791
2792 010012 023737 003210 003150 20$:    CMP      E.CS2,T.CS2    ;CHECK CS2 CORRECT
2793 010020 001401          BEQ     21$            ;YES, CONTINUE
2794 010022 104047          ERROR   47              ;CS2 INCORRECT
2795 010024 023737 003214 003154 21$:    CMP      E.ER,T.ER      ;CHECK ERROR REG CORRECT
  
```

```

2796 010032 001401 BEQ 22$ ;YES, CONTINUE
2797 010034 104050 ERROR 50 ;ERROR REG INCORRECT
2798 010036 023737 003222 003162 22$: CMP E.DB,T.DB ;CHECK DATA CORRECT
2799 010044 001401 BEQ 25$ ;YES, CONTINUE
2800 010046 104051 ERROR 51 ;DATA INCORRECT
2801 010050 005237 003264 25$: INC WRDCNT ;INCREMENT WORD COUNT
2802 010054 005301 DEC R1
2803 010056 001003 BNE 26$ ;CHECK IF LAST WORD
2804 010060 012737 000100 003210 MOV #IR,E.CS2 ;UPDATE EXPECTED CS2
2805 010066 100316 26$: BPL 15$ ;CHECK IF FINISHED
    
```

.SBTTL \*\*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.

\*\*\*\*\*

```

TST14: SCOPE
MOV #10, $TIMES ;DO 10. ITERATIONS
MOV $BASE, R2 ;LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ;CLEAR RK611
MOV #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #-1, RKWC(R2) ;WORD COUNT=1
MOV #BUFF, RKBA(R2) ;LOAD DUMMY BUS ADDRESS
MOV #WRDATA, RKCS1(R2) ;ISSUE WRITE DATA
MOV #69.*4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
; FOR SECTOR PULSE
2836 010146 012762 000440 000026 1$: MOV #DMD!MCLK, RKMR1(R2)
2837 010154 012762 000040 000026 MOV #DMD, RKMR1(R2)
2838 010162 005300 DEC R0
2839 010164 001370 BNE 1$
2840 010166 012762 000140 000026 MOV #DMD!MSP, RKMR1(R2) ;SIMULATE SECTOR PULSE
2841 010174 012762 000040 000026 MOV #DMD, RKMR1(R2)
2842 010202 012737 000023 003200 MOV #WRDATA, E.CS1 ;STORE EXPECTED CS1
2843 010210 005037 003254 CLR PR.BIT ;GENERATE SYNCH
2844 010214 005037 003256 CLR M1.BIT
2845 010220 005037 003262 CLR BITCNT
2846 010224 012700 000377 MOV #255., R0
2847 010230 004737 035422 5$: JSR PC, RDBIT
2848 010234 016237 000000 003140 MOV RKCS1(R2), T.CS1
2849 010242 023737 003200 003140 CMP E.CS1, T.CS1 ;CHECK CS1 CORRECT
2850 010250 001112 BNE 63$ ;NO, REPORT ERROR
2851 010252 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
    
```

```
2852 010256 005300          DEC      RO          ;CHECK IF SYNCH FINISHED
2853 010260 001363          BNE      5$
2854 010262 012737 000001 003254  MOV     #1,PR.BIT    ;SIMULATE SYNCH BIT
2855 010270 004737 035422          JSR     PC,RDBIT
2856 010274 016237 000000 003140  MOV     RKCS1(R2),T.CS1
2857 010302 023737 003200 003140  CMP     E.CS1,T.CS1  ;CHECK CS1 CORRECT
2858 010310 001072          BNE     63$          ;NO, REPORT CS1
2859 010312 005237 003262          INC     BITCNT       ;INCREMENT BIT COUNT
2860 010316 012703 051754          MOV     #HEAD1,R3    ;SIMULATE GOOD HEADER
2861 010322 012701 000003          MOV     #3,R1
2862 010326 012304          12$:   MOV     (R3)+,R4      ;GET NEXT HEADER WORD
2863 010330 012700 000020          MOV     #16.,RO      ;LOAD BITS PER WORD
2864 010334 013737 003254 003256 15$:   MOV     PR.BIT,M1.BIT ;STORE PREVIOUS BIT
2865 010342 006004          ROR     R4           ;GET NEXT BIT
2866 010344 103403          BCS     17$
2867 010346 005037 003254          CLR     PR.BIT
2868 010352 000403          BR      18$
2869
2870 010354 012737 000001 003254 17$:   MOV     #1,PR.BIT
2871 010362 004737 035422          18$:   JSR     PC,RDBIT    ;SIMULATE NEXT BIT
2872 010366 016237 000000 003140  MOV     RKCS1(R2),T.CS1 ;GET CS1
2873 010374 023737 003200 003140  CMP     E.CS1,T.CS1  ;CHECK CS1 CORRECT
2874 010402 001035          BNE     63$          ;NO, REPORT ERROR
2875 010404 005237 003262          INC     BITCNT       ;INCREMENT BIT COUNT
2876 010410 005300          DEC     RO           ;CHECK IF READY FOR NEXT HEADER WORD
2877 010412 001350          BNE     15$          ;NO, CONTINUE
2878 010414 005301          DEC     R1           ;CHECK IF FINISHED WITH HEAD
2879 010416 001343          BNE     12$          ;NO, CONTINUE
2880 010420 012700 000105          MOV     #69.,RO      ;LOAD COUNT FOR GAP
2881 010424 013737 003254 003256 25$:   MOV     PR.BIT,M1.BIT ;SIMULATE GAP
2882 010432 005037 003254          CLR     PR.BIT
2883 010436 004737 035422          JSR     PC,RDBIT
2884 010442 005300          DEC     RO
2885 010444 001367          BNE     25$
2886 010446 016237 000026 003164  MOV     RKMR1(R2),T.MR1 ;GET MR1
2887 010454 012737 062040 003224  MOV     #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;LOAD EXPECTED MR1
2888 010462 023737 003224 003164  CMP     E.MR1,T.MR1  ;CHECK IF WRITE GATE SET
2889 010470 001411          BEQ     TST15        ;:YES, GO ON TO NEXT TEST
2890 010472 104053          ERROR   53
2891 010474 000407          BR      TST15        ;:GO ON TO NEXT TEST
2892
2893 010476 016237 000010 003150 63$:   MOV     RKCS2(R2),T.CS2 ;STORE CS2 FOR PRINT OUT
2894 010504 016237 000014 003154  MOV     RKER(R2),T.ER  ;STORE ERROR REG FOR PRINT OUT
2895 010512 104052          ERROR   52          ;REPORT ERROR
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
```

```
*****
*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
```

```
2908          : *          140000
2909          : *          140000
2910          : *
2911          : *          MAKE SURE WRITE GATE DOES NOT SET.
2912          : *
2913          : *
2914 010514 000004 TST15: SCOPE
2915 010516 012737 000012 001200 MOV #10.,$TIMES ;;DO 10. ITERATIONS
2916 010524 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2917 010530 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2918 010536 012700 000700 MOV #700,R0 ;SET STALL COUNT
2919 010542 005300 4$: DEC RO ;DEC COUNT
2920 010544 001376 BNE 4$ ;LOOP UNTIL ZERO
2921 010546 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2922 010554 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT = 1
2923 010562 012762 053606 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
2924 010570 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
2925 010576 012700 000426 MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTILL READY
2926          : ; FOR SECTOR PULSE
2927 010602 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
2928 010610 012762 000040 000026 MOV #DMD,RKMR1(R2)
2929 010616 005300 DEC RO
2930 010620 001370 BNE 1$
2931 010622 012700 000004 MOV #4,R0 ;SIMULATE INDEX PULSE
2932 010626 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
2933 010634 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
2934 010642 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
2935 010650 005300 DEC RO
2936 010652 001370 BNE 2$
2937 010654 012762 000040 000026 MOV #DMD,RKMR1(R2)
2938 010662 005037 003254 CLR PR.BIT ;GENERATE SYNCH
2939 010666 005037 003256 CLR M1.BIT
2940 010672 012700 000377 MOV #255.,R0
2941 010676 004737 035422 5$: JSR PC,RDBIT
2942 010702 005300 DEC RO
2943 010704 001374 BNE 5$
2944 010706 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
2945 010714 004737 035422 JSR PC,RDBIT
2946 010720 012703 051754 MOV #HEAD1,R3 ;SIMULATE GOOD HEADER
2947 010724 012701 000003 MOV #3,R1
2948 010730 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
2949 010732 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORDS
2950 010736 013737 003254 003256 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
2951 010744 006004 ROR R4 ;GET NEXT BIT
2952 010746 103403 BCS 17$
2953 010750 005037 003254 CLR PR.BIT
2954 010754 000403 BR 18$
2955
2956 010756 012737 000001 003254 17$: MOV #1,PR.BIT
2957 010764 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
2958 010770 005300 DEC RO ;CHECK IF READY FOR NEXT HEADER WORD
2959 010772 001361 BNE 15$ ;NO,CONTINUE
2960 010774 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
2961 010776 001354 BNE 12$ ;NO,CONTINUE
2962 011000 012700 000100 MOV #64.,R0 ;LOAD COUNT FOR GAP
2963 011004 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
```



```

2964 011012 005037 003254 CLR PR.BIT
2965 011016 004737 035422 JSR PC,RDBIT
2966 011022 005300 DEC R0
2967 011024 001367 BNE 25$
2968 011026 016237 000026 003164 MOV RKMR1(R2),T.MR1 ;GET MR1
2969 011034 012737 022040 003224 MOV #DMD!ECCW!MEWD,E.MR1 ;LOAD EXPECTDED MR1
2970 011042 023737 003164 003224 CMP T.MR1,E.MR1 ;CHECK IF WRITE GATE DID NOT SET
2971 011050 001401 BEQ TST16 ;:YES, GO ON TO NEXT TEST
2972 011052 104053 ERROR 53
  
```

```

*****
:TEST 16 SECTOR 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 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.
*****
  
```

```

2986
2987 011054 000004 TST16: SCOPE
2988 011056 012737 000012 001200 MOV #10,$TIMES ;:DO 10. ITERATIONS
2989 011064 013702 001270 MOV $BASE,R2 ;:LOAD RK611 BASE
2990 011070 005037 003300 CLR CYLN ;:INITIALIZE CYLINDER
2991 011074 005037 003276 CLR SECTOR ;:INITIALIZE TRACK AND SECTOR
2992 011100 012737 011106 001110 MOV #1$,$LPERR ;:LOAD LOOP ON ERROR LOCATION FOR
2993 ; SUBTEST LOOP
2994
2995 011106 1$: JSR PC,INCHDR ;:GENERATE HEADER WORDS
2996 011106 004737 034220 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611 CONTROLLER
2997 011112 012762 100000 000000 MOV #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
2998 011120 012762 000040 000026 MOV CYLN,RKDCYL(R2) ;:LOAD DESIRED CYLINDER
2999 011126 013762 003300 000020 MOV SECTOR,RKDA(R2) ;:LOAD DESIRED TRACK/SECTOR
3000 011134 013762 003276 000006 MOV #-1,RKWC(R2) ;:WORD COUNT=1
3001 011142 012762 177777 000002 MOV #BUFF,RKBA(R2) ;:LOAD DUMMY ADDRESS
3002 011150 012762 053606 000004 MOV #WRDATA,RKCS1(R2) ;:ISSUE WRITE DATA
3003 011156 012762 000023 000000 MOV #69.*4+2,R0 ;:ISSUE ENOUGH CLOCKS UNTIL READY
3004 011164 012700 000426 ; FOR SECTOR PULSE
3005
3006 011170 012762 000440 000026 5$: MOV #DMD!MCLK,RKMR1(R2)
3007 011176 012762 000040 000026 MOV #DMD,RKMR1(R2)
3008 011204 005300 DEC R0
3009 011206 001370 BNE 5$
3010 011210 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;:SIMULATE SECTOR PULSE
3011 011216 012762 000040 000026 MOV #DMD,RKMR1(R2)
3012 011224 005037 003254 CLR PR.BIT ;:GENERATE SYNCH
3013 011230 005037 003256 CLR M1.BIT
3014 011234 012700 000377 MOV #255.,R0
3015 011240 004737 035422 10$: JSR PC,RDBIT
3016 011244 005300 DEC R0
3017 011246 001374 BNE 10$
3018 011250 012737 000001 003254 MOV #1,PR.BIT ;:SIMULATE SYNCH BIT
3019 011256 004737 035422 JSR PC,RDBIT
  
```



```

3076 011502          1$:
3077 011502 004737 034220 JSR PC,INCHDR ;GENERATE HEADER WORDS
3078 011506 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611 CONTROLLER
3079 011514 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3080 011522 013762 003300 000020 MOV CYLN,RKDCYL(R2) ;LOAD DESIRED CYLINDER
3081 011530 013762 003276 000006 MOV SECTOR,RKDA(R2) ;LOAD DESIRED TRACK/SECTOR
3082 011536 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT=1
3083 011544 012762 053606 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY ADDRESS
3084 011552 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3085 011560 012700 000426 MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3086                                     ; FOR SECTOR PULSE
3087 011564 012762 000440 000026 5$: MOV #DMD!MCLK,RKMR1(R2)
3088 011572 012762 000040 000026 MOV #DMD,RKMR1(R2)
3089 011600 005300 DEC R0
3090 011602 001370 BNE 5$
3091 011604 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3092 011612 012762 000040 000026 MOV #DMD,RKMR1(R2)
3093 011620 005037 003254 CLR PR.BIT ;GENERATE SYNCH
3094 011624 005037 003256 CLR M1.BIT
3095 011630 012700 000377 MOV #255.,R0
3096 011634 004737 035422 10$: JSR PC,RDBIT
3097 011640 005300 DEC R0
3098 011642 001374 BNE 10$
3099 011644 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
3100 011652 004737 035422 JSR PC,RDBIT
3101 011656 012703 003302 MOV #HEADER,R3 ;SIMULATE HEADER
3102 011662 012701 000003 MOV #3,R1
3103 011666 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
3104 011670 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
3105 011674 013737 003254 003256 15$: MOV PR.BIT,M1.BIT
3106 011702 006004 ROR R4
3107 011704 103403 BCS 17$
3108 011706 005037 003254 CLR PR.BIT
3109 011712 000403 BR 18$
3110
3111 011714 012737 000001 003254 17$: MOV #1,PR.BIT
3112 011722 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
3113 011726 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
3114 011730 001361 BNE 15$ ;NO, CONTINUE
3115 011732 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
3116 011734 001354 BNE 12$ ;NO, CONTINUE
3117 011736 012700 000100 MOV #64.,R0 ;LOAD COUNT FOR GAP
3118 011742 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3119 011750 005037 003254 CLR PR.BIT
3120 011754 004737 035422 JSR PC,RDBIT
3121 011760 005300 DEC R0 ;CHECK IF GAP FINISHED
3122 011762 001367 BNE 25$ ;NO, CONTINUE
3123 011764 016237 000006 003146 MOV RKDA(R2),T.DA ;GET DISK ADDRESS REG
3124 011772 016237 000020 003160 MOV RKDCYL(R2),T.DCYL ;GET CYLINDER ADD REG.
3125 012000 023737 003206 003146 CMP E.DA,T.DA ;CHECK DISK ADD CORRECT
3126 012006 001401 BEQ 30$ ;YES, CONTINUE
3127 012010 104056 ERROR 56 ;DISK ADDRESS INCORRECT
3128 012012 023737 003220 003160 30$: CMP E.DCYL,T.DCYL ;CHECK IF CYLINDER ADD CORRECT
3129 012020 001401 BEQ 32$ ;YES, CHECK IF LOOP ON ERROR
3130 012022 104002 ERROR R2 ;CYLINDER INCORRECT
3131 012024 104415 32$: SCOP1 ;CHECK IF LOOP ON ERROR
  
```

3132 012026 105237 003277  
 3133 012032 122737 000003 003277  
 3134 012040 001220  
 3135  
 3136  
 3137  
 3138  
 3139  
 3140  
 3141  
 3142  
 3143  
 3144  
 3145  
 3146  
 3147  
 3148  
 3149 012042 000004  
 3150 012044 012737 000012 001200  
 3151 012052 013702 001270  
 3152 012056 005037 003300  
 3153 012062 012737 001025 003276  
 3154 012070 012737 012076 001110  
 3155  
 3156  
 3157 012076  
 3158 012076 004737 034220  
 3159 012102 012762 100000 000000  
 3160 012110 012762 000040 000026  
 3161 012116 013762 003300 000020  
 3162 012124 013762 003276 000006  
 3163 012132 012762 177777 000002  
 3164 012140 012762 053606 000004  
 3165 012146 012762 000023 000000  
 3166 012154 012700 000426  
 3167  
 3168 012160 012762 000440 000026 5\$:  
 3169 012166 012762 000040 000026  
 3170 012174 005300  
 3171 012176 001370  
 3172 012200 012762 000140 000026  
 3173 012206 012762 000040 000026  
 3174 012214 005037 003254  
 3175 012220 005037 003256  
 3176 012224 012700 000377  
 3177 012230 004737 035422 10\$:  
 3178 012234 005300  
 3179 012236 001374  
 3180 012240 012737 000001 003254  
 3181 012246 004737 035422  
 3182 012252 012703 003302  
 3183 012256 012701 000003  
 3184 012262 012304 12\$:  
 3185 012264 012700 000020  
 3186 012270 013737 003254 003256 15\$:  
 3187 012276 006004

INCB TRACK :INCREMENT TRACK  
 CMPB #3,TRACK :CHECK IF ALL TRACKS TRIED  
 BNE 1\$ :NO, TRY NEXT VALUE

\*\*\*\*\*  
 :TEST 20 CYLINDER 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 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.  
 :  
 :\*\*\*\*\*  
 TST20: SCOPE  
 MOV #10, \$TIMES ;;DO 10. ITERATIONS  
 MOV \$BASE, R2 ;LOAD RK611 BASE  
 CLR CYLN ;INITIALIZE CYLINDER  
 MOV #1025, 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

12\$:  
 MOV (R3)+, R4 ;GET NEXT HEADER WORD  
 MOV #16., R0 ;LOAD BITS PER WORD

15\$:  
 MOV PR.BIT, M1.BIT  
 ROR R4

```

3188 012300 103403          BCS      17$
3189 012302 005037 003254  CLR      PR.BIT
3190 012306 000403          BR       18$
3191
3192 012310 012737 000001 003254 17$:  MOV     #1,PR.BIT
3193 012316 004737 035422 18$:  JSR     PC,RDBIT      ;SIMULATE NEXT BIT
3194 012322 005300          DEC     R0           ;CHECK IF READY FOR NEXT HEADER WORD
3195 012324 001361          BNE     15$         ;NO, CONTINUE
3196 012326 005301          DEC     R1           ;CHECK IF FINISHED WITH HEADER
3197 012330 001354          BNE     12$         ;NO, CONTINUE
3198 012332 012700 000100          MOV     #64.,R0     ;LOAD COUNT FOR GAP
3199 012336 013737 003254 003256 25$:  MOV     PR.BIT,M1.BIT ;SIMULATE GAP
3200 012344 005037 003254          CLR     PR.BIT
3201 012350 004737 035422          JSR     PC,RDBIT
3202 012354 005300          DEC     R0           ;CHECK IF GAP FINISHED
3203 012356 001367          BNE     25$         ;NO, CONTINUE
3204 012360 016237 000006 003146          MOV     RKDA(R2),T.DA ;GET DISK ADDRESS REG
3205 012366 016237 000020 003160          MOV     RKDCYL(R2),T.DCYL ;GET CYLINDER ADD REG.
3206 012374 023737 003206 003146          CMP     E.DA,T.DA   ;CHECK DISK ADD CORRECT
3207 012402 001401          BEQ     30$         ;YES, CONTINUE
3208 012404 104060          ERROR  60          ;DISK ADDRESS INCORRECT
3209 012406 023737 003220 003160 30$:  CMP     E.DCYL,T.DCYL ;CHECK IF CYLINDER ADD CORRECT
3210 012414 001401          BEQ     32$         ;YES, CHECK IF LOOP ON ERROR
3211 012416 104002          ERROR  R2          ;CYLINDER INCORRECT
3212 012420 104415          32$:  SCOP1          ;CHECK IF LOOP ON ERROR
3213 012422 005237 003300          INC     CYLN        ;INCREMENT CYLINDER
3214 012426 022737 000633 003300          CMP     #633,CYLN   ;CHECK IF ALL CYLINDER TRIED
3215 012434 001220          BNE     1$          ;NO, TRY NEXT VALUE
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
    
```

```

*****
: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 ADDRESS
:   IS NOT INCREMENTED.
:
*****
    
```

```

TST21:  SCOPE
3235 012436 000004          MOV     #10.,$TIMES  ;:DO 10. ITERATIONS
3236 012440 012737 000012 001200          MOV     $BASE,R2    ;:LOAD RK611 BASE
3237 012446 013702 001270          MOV     #CCLR,RKCS1(R2) ;:CLEAR RK611
3238 012452 012762 100000 000000          MOV     #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
3239 012460 012762 000040 000026          MOV     #0,RKDCYL(R2) ;:LOAD CYLINDER
3240 012466 012762 000000 000020          MOV     #0,RKDA(R2)  ;:LOAD TRACK AND SECTOR
3241 012474 012762 000000 000006          MOV     #-1,RKWC(R2) ;:WORD COUNT=1
3242 012502 012762 177777 000002          MOV     #BUFF,RKBA(R2) ;:LOAD DUMMY BUS ADDRESS
3243 012510 012762 053606 000004
    
```

```
3244 012516 012762 000023 000000      MOV      #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3245 012524 012700 000426      MOV      #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3246                                     ; FOR SECTOR PULSE
3247 012530 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2)
3248 012536 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3249 012544 005300      DEC      R0
3250 012546 001370      BNE     1$
3251 012550 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3252 012556 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3253 012564 005037 003254      CLR     PR.BIT ;GENERATE SYNCH
3254 012570 005037 003256      CLR     M1.BIT
3255 012574 012700 000377      MOV      #255.,R0
3256 012600 004737 035422      JSR     PC,RDBIT
3257 012604 005300      DEC     R0 ;CHECK IF SYNCH FINISHED
3258 012606 001374      BNE     5$
3259 012610 012737 000001 003254      MOV      #1,PR.BIT ;SIMULATE SYNCH BIT
3260 012616 004737 035422      JSR     PC,RDBIT
3261 012622 012703 051762      MOV      #HEAD2,R3 ;SIMULATE HEADER ERROR
3262 012626 012701 000003      MOV      #3,R1
3263 012632 012304      MOV      (R3)+,R4 ;GET NEXT HEADER WORD
3264 012634 012700 000020      MOV      #16.,R0 ;LOAD BITS PER WORD
3265 012640 013737 003254 003256 15$:  MOV      PR.BIT,M1.BIT ;STORE PREVIOUS BIT
3266 012646 006004      ROR     R4 ;GET NEXT BIT
3267 012650 103403      BCS     17$
3268 012652 005037 003254      CLR     PR.BIT
3269 012656 000403      BR      18$
3270
3271 012660 012737 000001 003254 17$:  MOV      #1,PR.BIT
3272 012666 004737 035422 18$:  JSR     PC,RDBIT ;SIMULATE NEXT BIT
3273 012672 005300      DEC     R0 ;CHECK IF READY FOR NEXT HEADER WORD
3274 012674 001361      BNE     15$ ;NO, CONTINUE
3275 012676 005301      DEC     R1 ;CHECK IF FINISHED WITH HEAD2
3276 012700 001354      BNE     12$ ;NO, CONTINUE
3277 012702 012700 000100      MOV      #64.,R0 ;LOAD COUNT FOR GAP
3278 012706 013737 003254 003256 25$:  MOV      PR.BIT,M1.BIT ;SIMULATE GAP
3279 012714 005037 003254      CLR     PR.BIT
3280 012720 004737 035422      JSR     PC,RDBIT
3281 012724 005300      DEC     R0 ;CHECK IF GAP FINISHED
3282 012726 001367      BNE     25$ ;NO, CONTINUE
3283 012730 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;GET CS1
3284 012736 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;GET CS2
3285 012744 016237 000014 003154      MOV      RKER(R2),T.ER ;GET ERROR REG
3286 012752 012737 000023 003200      MOV      #WRDATA,E.CS1 ;LOAD EXPECTED CS1
3287 012760 012737 000300 003210      MOV      #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3288 012766 012737 000200 003214      MOV      #BSE,E.ER ;LOAD EXPECTED ERROR REG.
3289 012774 023737 003200 003140      CMP     E.CS1,T.CS1 ;CHECK CS1 CORRECT
3290 013002 001401      BEQ     30$ ;YES, CHECK CS2
3291 013004 104063      ERROR  63 ;CS1 INCORRECT
3292 013006 023737 003210 003150 30$:  CMP     E.CS2,T.CS2 ;CHECK CS2 CORRECT
3293 013014 001401      BEQ     32$ ;YES, CHECK ERROR REG.
3294 013016 104064      ERROR  64 ;CS2 INCORRECT
3295 013020 023737 003214 003154 32$:  CMP     E.ER,T.ER ;CHECK ERROR REG CORRECT
3296 013026 001401      BEQ     TST22 ;:YES, GO ON TO NEXT TEST
3297 013030 104065      ERROR  65 ;ERROR REG INCORRECT
3298
3299 ;:*****
```

```

3300 ;*TEST 22      BAD SECTOR ERROR (PART 2)
3301 ;*
3302 ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.  PUT
3303 ;*      CONTROLLER IN MAINTENANCE MODE.  ISSUE A WRITE DATA OF
3304 ;*      ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0,
3305 ;*      HEAD 0, SECTOR 0.  CLOCK IN BOTH SEEK AND DRIVE CLEAR
3306 ;*      MESSAGES.  SIMULATE A SECTOR PULSE AND A HEADER WITH
3307 ;*      THE FOLLOWING DATA:
3308 ;*
3309 ;*      000000
3310 ;*      100000
3311 ;*      100000
3312 ;*
3313 ;*      MAKE SURE BAD SECTOR ERROR SETS.  CHECK THAT DISK ADDRESS
3314 ;*      IS NOT INCREMENTED.
3315 ;*
3316 ;*
3317 ;*.....
3317 013032 000004 TST22: SCOPE
3318 013034 012737 000012 001200 MOV #10.,$TIMES ;:DO 10. ITERATIONS
3319 013042 013702 001270 MOV $BASE,R2 ;:LOAD RK611 BASE
3320 013046 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
3321 013054 012762 000040 000026 MOV #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
3322 013062 012762 000000 000020 MOV #0,RKDCYL(R2) ;:LOAD CYLINDER
3323 013070 012762 000000 000006 MOV #0,RKDA(R2) ;:LOAD TRACK AND SECTOR
3324 013076 012762 177777 000002 MOV #-1,RKWC(R2) ;:WORD COUNT=1
3325 013104 012762 053606 000004 MOV #BUFF,RKBA(R2) ;:LOAD DUMMY BUS ADDRESS
3326 013112 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;:ISSUE WRITE DATA
3327 013120 012700 000426 MOV #69.*4+2,R0 ;:ISSUE ENOUGH CLOCKS UNTIL READY
3328 ;: FOR SECTOR PULSE
3329 013124 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
3330 013132 012762 000040 000026 MOV #DMD,RKMR1(R2)
3331 013140 005300 DEC R0
3332 013142 001370 BNE 1$
3333 013144 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;:SIMULATE SECTOR PULSE
3334 013152 012762 000040 000026 MOV #DMD,RKMR1(R2)
3335 013160 005037 003254 CLR PR.BIT ;:GENERATE SYNCH
3336 013164 005037 003256 CLR M1.BIT
3337 013170 012700 000377 MOV #255.,R0
3338 013174 004737 035422 5$: JSR PC,RDBIT
3339 013200 005300 DEC R0 ;:CHECK IF SYNCH FINISHED
3340 013202 001374 BNE 5$
3341 013204 012737 000001 003254 MOV #1,PR.BIT ;:SIMULATE SYNCH BIT
3342 013212 004737 035422 JSR PC,RDBIT
3343 013216 012703 051770 MOV #HEAD3,R3 ;:SIMULATE HEADER ERROR
3344 013222 012701 000003 MOV #3,R1
3345 013226 012304 12$: MOV (R3)+,R4 ;:GET NEXT HEADER WORD
3346 013230 012700 000020 MOV #16.,R0 ;:LOAD BITS PER WORD
3347 013234 013737 003254 15$: MOV PR.BIT,M1.BIT ;:STORE PREVIOUS BIT
3348 013242 006004 ROR R4 ;:GET NEXT BIT
3349 013244 103403 BCS 17$
3350 013246 005037 003254 CLR PR.BIT
3351 013252 000403 BR 18$
3352 ;
3353 013254 012737 000001 003254 17$: MOV #1,PR.BIT
3354 013262 004737 035422 18$: JSR PC,RDBIT ;:SIMULATE NEXT BIT
3355 013266 005300 DEC R0 ;:CHECK IF READY FOR NEXT HEADER WORD

```

```
3356 013270 001361 BNE 15% ;NO, CONTINUE
3357 013272 005301 DEC R1 ;CHECK IF FINISHED WITH HEAD3
3358 013274 001354 BNE 12% ;NO, CONTINUE
3359 013276 012700 000100 MOV #64,R0 ;LOAD COUNT FOR GAP
3360 013302 013737 003254 003256 25%: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3361 013310 005037 003254 CLR PR.BIT
3362 013314 004737 035422 JSR PC,RDBIT
3363 013320 005300 DEC R0 ;CHECK IF GAP FINISHED
3364 013322 001367 BNE 25% ;NO, CONTINUE
3365 013324 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
3366 013332 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
3367 013340 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
3368 013346 012737 000023 003200 MOV #WRDATA,E.CS1 ;LOAD EXPECTED CS1
3369 013354 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3370 013362 012737 000200 003214 MOV #BSE,E.ER ;LOAD EXPECTED ERROR REG.
3371 013370 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
3372 013376 001401 BEQ 30% ;YES, CHECK CS2
3373 013400 104063 ERROR 63 ;CS1 INCORRECT
3374 013402 023737 003210 003150 30%: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
3375 013410 001401 BEQ 32% ;YES, CHECK ERROR REG.
3376 013412 104064 ERROR 64 ;CS2 INCORRECT
3377 013414 023737 003214 003154 32%: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
3378 013422 001401 BEQ TST23 ;:YES, GO ON TO NEXT TEST
3379 013424 104065 ERROR 65 ;ERROR REG INCORRECT
```

```
3380
3381
3382 :*****
3383 :*TEST 23 OPERATION INCOMPLETE
3384 :*
3385 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3386 :* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
3387 :* ONE WORD TO AN RK06 IN 24 SECTOR FORMAT, CYLINDER 1253,
3388 :* HEAD 2, SECTOR 23. CLOCK IN BOTH SEEK AND DRIVE CLEAR
3389 :* MESSAGES. SIMULATE A SECTOR PULSE AND 32 SECTORS WITH
3390 :* 1 BIT DIFFERENT IN 30 BITS OF OPI DETERMINATION. ALL
3391 :* SIMULATED HEADERS, HAVE GOOD HEADER VRC. MAKE SURE ONLY
3392 :* OPERATION INCOMPLETE AND CONTROLLER ARE THE ONLY ERRORS
3393 :* THAT SET.
3394 :*****
```

```
3395 013426 000004 TST23: SCOPE
3396 013430 012737 000012 001200 MOV #10,$TIMES ;;DO 10. ITERATIONS
3397 013436 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
3398 013442 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
3399 013450 012700 002500 MOV #2500,R0 ;SET COUNT FOR STALL
3400 013454 005300 2%: DEC R0 ;DEC COUNT
3401 013456 001376 BNE 2% ;LOOP UNTIL 0
3402 013460 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3403 013466 012762 053606 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUFFER ADDRESS
3404 013474 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT = 1
3405 013502 012762 001253 000020 MOV #1253,RKDCYL(R2) ;LOAD CYLINDER
3406 013510 012762 001023 000006 MOV #1023,RKDA(R2) ;LOAD TRACK AND SECTOR
3407 013516 012762 010023 000000 MOV #WRDATA!CFMT,RKCS1(R2) ;ISSUE WRITE DATA
3408 013524 012700 000426 MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3409 : ; FOR SECTOR PULSE
3410 013530 012762 000440 000026 1%: MOV #DMD!MCLK,RKMR1(R2)
3411 013536 012762 000040 000026 MOV #DMD,RKMR1(R2)
```



3412	013544	005300			DEC	R0	
3413	013546	001370			BNE	18	
3414	013550	012705	000040		MOV	#32.,R5	:LOAD HEADER COUNT
3415	013554	012703	052164		MOV	#OPI1,R3	:LOAD ADDRESS OF HEADERS
3416	013560	012737	010023	003200	MOV	#WRDATA!CFMT,E.CS1	:LOAD EXPECTED CS1
3417	013566	012737	000300	003210	MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3418	013574	005037	003214		CLR	E.ER	:LOAD EXPECTED ERROR REG
3419	013600	012737	022040	003224	MOV	#DMD!MEWD!ECCW,E.MR1	:LOAD EXPECTED MR1
3420	013606	005037	003310		CLR	HDRCNT	:INITIALIZE HEADER COUNT
3421	013612	012762	000140	000026	5\$: MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
3422	013620	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3423	013626	022737	000037	003310	CMP	#31.,HDRCNT	:CHECK IF ALL HEADERS DONE
3424	013634	001460			BEQ	26\$	:YES - SKIP TO ERROR TEST
3425	013636	005037	003254		CLR	PR.BIT	:GENERATE SYNCH
3426	013642	005037	003256		CLR	M1.BIT	
3427	013646	012700	000377		MOV	#255.,R0	
3428	013652	004737	035422		10\$: JSR	PC,RDBIT	
3429	013656	005300			DEC	R0	:CHECK IF SYNCH FINISHED
3430	013660	001374			BNE	10\$	
3431	013662	012737	000001	003254	MOV	#1,PR.BIT	:SIMULATE SYNCH BIT
3432	013670	004737	035422		JSR	PC,RDBIT	
3433	013674	012701	000003		MOV	#3,R1	:SIMULATE OPI
3434	013700	012304			12\$: MOV	(R3)+,R4	:GET NEXT HEADER WORD
3435	013702	012700	000020		MOV	#16.,R0	:LOAD BITS PER WORD
3436	013706	013737	003254	003256	15\$: MOV	PR.BIT,M1.BIT	:STORE PREVIOUS BIT
3437	013714	006004			ROR	R4	:GET NEXT BIT
3438	013716	103403			BCS	17\$	
3439	013720	005037	003254		CLR	PR.BIT	
3440	013724	000403			BR	18\$	
3441							
3442	013726	012737	000001	003254	17\$: MOV	#1,PR.BIT	
3443	013734	004737	035422		18\$: JSR	PC,RDBIT	:SIMULATE NEXT BIT
3444	013740	005300			DEC	R0	:CHECK IF READY FOR NEXT HEADER WORD
3445	013742	001361			BNE	15\$	:NO, CONTINUE
3446	013744	005301			DEC	R1	:CHECK IF FINISHED WITH HEADER
3447	013746	001354			BNE	12\$	:NO,CONTINUE
3448	013750	012700	000100		MOV	#64.,R0	:LOAD COUNT FOR GAP
3449	013754	013737	003254	003256	25\$: MOV	PR.BIT,M1.BIT	:SIMULATE GAP
3450	013762	005037	003254		CLR	PR.BIT	
3451	013766	004737	035422		JSR	PC,RDBIT	
3452	013772	005300			DEC	R0	:CHECK IF GAP IS FINISHED
3453	013774	001367			BNE	25\$	:NO, CONTINUE
3454	013776	016237	000000	003140	26\$: MOV	RKCS1(R2),T.CS1	:GET CS1
3455	014004	016237	000010	003150	MOV	RKCS2(R2),T.CS2	:GET CS2
3456	014012	016237	000014	003154	MOV	RKER(R2),T.ER	:GET ERROR REG.
3457	014020	023737	003200	003140	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
3458	014026	001401			BEQ	30\$	:YES, CONTINUE
3459	014030	104074			ERROR	74	:CS1 INCORRECT
3460	014032	023737	003210	003150	30\$: CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
3461	014040	001401			BEQ	31\$	:YES, CONTINUE
3462	014042	104075			ERROR	75	:CS2 INCORRECT
3463	014044	023737	003214	003154	31\$: CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
3464	014052	001401			BEQ	32\$	:YES, CONTINUE
3465	014054	104076			ERROR	76	:ERROR REG INCORRECT
3466	014056	016237	000026	003164	32\$: MOV	RKMR1(R2),T.MR1	:GET MR1
3467	014064	023737	003224	003164	CMP	E.MR1,T.MR1	:CHECK TO MAKE SURE WRITE GATE DID NOT SET

3468	014072	001401				BEQ	34\$	:YES, CONTINUE
3469	014074	104077				ERROR	77	:MR1 INCORRECT
3470	014076	005237	003310		34\$:	INC	HDRCNT	:INCREMENT HEADER COUNT
3471	014102	022737	000037	003310		CMP	#31.,HDRCNT	:CHECK IF LAST HEADER
3472	014110	001011				BNE	35\$	:NO, CHECK IF FINSHED
3473	014112	012737	020000	003214		MOV	#OPI,E.ER	:LOAD ERROR BIT
3474	014120	042737	000001	003200		BIC	#GO,E.CS1	:ADJUST E.CS1 FOR END OF OP CONTENTS
3475	014126	052737	100200	003200		BIS	#RDY!CERR,E.CS1	
3476	014134	005305			35\$:	DEC	R5	:CHECK IF FINISHED
3477	014136	001402				BEQ	37\$	:YES - SKIP
3478	014140	000137	013612			JMP	5\$	:DO NEXT SECTOR
3479	014144	012762	100000	000000	37\$:	MOV	#CCLR,RKCS1(R2)	:CLEAR CONTROLLER
3480	014152	016237	000000	003140		MOV	RKCS1(R2),T.CS1	:GET CS1
3481	014160	016237	000010	003150		MOV	RKCS2(R2),T.CS2	:CS2
3482	014166	016237	000014	003154		MOV	RKER(R2),T.ER	:ER
3483	014174	012737	000200	003200		MOV	#RDY,E.CS1	:SET EXPECTED CS1
3484	014202	023737	003140	003200		CMP	T.CS1,E.CS1	:CHECK IF CORRECT
3485	014210	001401				BEQ	TST24	:GO TO NEXT TEST
3486	014212	104153				ERROR	153	

3487  
3488  
3489  
3490  
3491  
3492  
3493  
3494  
3495  
3496  
3497  
3498  
3499  
3500  
\*\*\*\*\*  
: \*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.  
: \*  
: \*  
\*\*\*\*\*

3501	014214	000004				TST24:	SCOPE	
3502	014216	012737	000012	001200		MOV	#10.,\$TIMES	::DO 10. ITERATIONS
3503	014224	013702	001270			MOV	\$BASE,R2	:LOAD RK611 BASE
3504	014230	012737	012023	003200		MOV	#CFMT!CDT!WRDATA,E.CS1	:LOAD EXPECTED CS1
3505	014236	012737	000300	003210		MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3506	014244	012737	000400	003214		MOV	#HVRC,E.ER	:LOAD EXPECTED ERROR REGISTER
3507	014252	012737	000001	001170		MOV	#1,\$TMP4	:INITIALIZE BIT FOR VRC ERROR
3508	014260	012737	014266	001110		MOV	#1\$,\$LPERR	:LOAD LOOP ON ERROR LOCATION FOR : SUBTEST LOOP
3509								
3510								
3511	014266				1\$:	MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
3512	014266	012762	100000	000000		MOV	#2000,R0	:SET COUNT FOR STALL
3513	014274	012700	002000					
3514	014300	005300			2\$:	DEC	R0	:DEC COUNT
3515	014302	001376				BNE	2\$	:LOOP UNTIL 0
3516	014304	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN MAINTENANCE MODE
3517	014312	012737	140370	053570		MOV	#140370,VRCHDR+4	:LOAD VRC
3518	014320	033737	001170	053570		BIT	\$TMP4,VRCHDR+4	:MAKE ONE BIT BAD
3519	014326	001404				BEQ	3\$	
3520	014330	043737	001170	053570		BIC	\$TMP4,VRCHDR+4	
3521	014336	000403				BR	5\$	
3522								
3523	014340	053737	001170	053570	3\$:	BIS	\$TMP4,VRCHDR+4	

```

3524 014346 012762 053606 000004 5$: MOV #BUFF,RKBA(R2) ;LOAD DUMMY ADDRESS
3525 014354 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT = 1
3526 014362 012762 001023 000006 MOV #1023,RKDA(R2) ;LOAD TRACK 2 SECTOR 23
3527 014370 012762 001253 000020 MOV #1253,RKDCYL(R2) ;LOAD CYLINDER 1253
3528 014376 012762 012023 000000 MOV #CFMT!CDT!WRDATA,RKCS1(R2) ;ISSUE COMMAND
3529 014404 012700 000426 MOV #69.*4+2,RO ;LOAD COUNT UNTIL READY FOR SECTOR
3530 014410 012762 000440 000026 10$: MOV #DMD!MCLK,RKMR1(R2)
3531 014416 012762 000040 000026 MOV #DMD,RKMR1(R2)
3532 014424 005300 DEC RO
3533 014426 001370 BNE 10$
3534 014430 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3535 014436 012762 000040 000026 MOV #DMD,RKMR1(R2)
3536 014444 005037 003254 CLR PR.BIT ;INITIAL BITS
3537 014450 005037 003256 CLR M1.BIT
3538 014454 012700 000377 MOV #255.,RO ;SIMULATE SYNCH
3539 014460 004737 035422 15$: JSR PC,RDBIT
3540 014464 005300 DEC RO
3541 014466 001374 BNE 15$
3542 014470 012737 000001 003254 MOV #1,PR.BIT ;GENERATE SYNCH BIT
3543 014476 004737 035422 JSR PC,RDBIT ;SIMULSTE SYNC 1 BIT
3544 014502 012703 053564 MOV #VRCHDR,R3 ;LOAD ADDRESS OF HEADER
3545 014506 012701 000003 MOV #3,R1 ;LOAD HEADER COUNT
3546 014512 012304 17$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
3547 014514 012700 000020 MOV #16.,RO ;LOAD BITS PER WORD
3548 014520 013737 003254 20$: MOV PR.BIT,M1.BIT ;SIMULATE NEXT BIT
3549 014526 006004 RCR R4
3550 014530 103403 BCS 23$
3551 014532 005037 003254 CLR PR.BIT
3552 014536 000403 BR 25$
3553
3554 014540 012737 000001 003254 23$: MOV #1,PR.BIT
3555 014546 004737 035422 25$: JSR PC,RDBIT
3556 014552 005300 DEC RO ;CHECK IF FINISHED WITH WORD
3557 014554 001361 BNE 20$ ;NO, CONTINUE
3558 014556 005301 DEC R1 ;CHECK IF HEADER FINISHED
3559 014560 001354 BNE 17$ ;NO, CONTINUE
3560 014562 012700 000100 MOV #64.,RO ;SIMULATE GAP
3561 014566 013737 003254 003256 30$: MOV PR.BIT,M1.BIT
3562 014574 005037 003254 CLR PR.BIT
3563 014600 004737 035422 JSR PC,RDBIT
3564 014604 005300 DEC RO
3565 014606 001367 BNE 30$
3566 014610 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;STORE CS1
3567 014616 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;STORE CS2
3568 014624 016237 000014 003154 MOV RKER(R2),T.ER ;STORE ERROR REG
3569 014632 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
3570 014640 001401 BEQ 35$ ;YES, CHECK CS2
3571 014642 104071 ERROR 71 ;CS1 INCORRECT
3572 014644 023737 003210 003150 35$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
3573 014652 001401 BEQ 37$ ;YES, CHECK ERROR REG
3574 014654 104072 ERROR 72 ;CS2 INCORRECT
3575 014656 023737 003214 003154 37$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
3576 014664 001401 BEQ 40$ ;YES, CHECK IF LOOP ON ERROR
3577 014666 104073 ERROR 73 ;ERROR REG INCORRECT
3578 014670 104415 40$: SCOP1 ;CHECK IF LOOP ON ERROR
3579 014672 006337 001170 ASL $TMP4 ;GET NEXT PATTERN

```

3580 014676 103402  
3581 014700 000137 014266

BCS TST25 ;:CHECK IF FINISHED  
JMP 1\$ ;NO, CONTINUE

```

3582
3583
3584 :*****
3585 :*TEST 25 BAD SECTOR ERROR AND HEADER VRC
3586 :*
3587 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
3588 :* CONTROLLER IN MAINTENANCE MODE. ISSUE A WRITE DATA OF
3589 :* ONE WORD TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 300.
3590 :* HEAD 1, SECTOR 17, CLOCK IN BOTH SEEK AND DRIVE CLEAR
3591 :* MESSAGES. SIMULATE THE FOLLOWING HEADER:
3592 :*
3593 :* 000300
3594 :* 040057
3595 :* 040356
3596 :*
3597 :* MAKE SURE ONLY HEADER VRC ERROR SETS.
3598 :*****

```

```

3599 014704 000004 TST25: SCOPE
3600 014706 012737 000012 001200 MOV #10.,$TIMES ;:DO 10. ITERATIONS
3601 014714 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
3602 014720 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
3603 014726 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3604 014734 012762 000300 000020 MOV #300,RKDCYL(R2) ;LOAD CYLINDER
3605 014742 012762 000417 000006 MOV #417,RKDA(R2) ;LOAD TRACK AND SECTOR
3606 014750 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT=1
3607 014756 012762 053606 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
3608 014764 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3609 014772 012700 000426 MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3610 ;: FOR SECTOR PULSE
3611 014776 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
3612 015004 012762 000040 000026 MOV #DMD,RKMR1(R2)
3613 015012 005300 DEC R0
3614 015014 001370 BNE 1$
3615 015016 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3616 015024 012762 000040 000026 MOV #DMD,RKMR1(R2)
3617 015032 005037 003254 CLR PR.BIT ;GENERATE SYNCH
3618 015036 005037 003256 CLR M1.BIT
3619 015042 012700 000377 MOV #255.,R0
3620 015046 004737 035422 5$: JSR PC,RDBIT
3621 015052 005300 DEC R0 ;CHECK IF SYNCH FINISHED
3622 015054 001374 BNE 5$
3623 015056 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
3624 015064 004737 035422 JSR PC,RDBIT
3625 015070 012703 051776 MOV #HEAD4,R3 ;SIMULATE HEADER ERROR
3626 015074 012701 000003 MOV #3,R1
3627 015100 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
3628 015102 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
3629 015106 013737 003254 003256 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
3630 015114 006004 ROR R4 ;GET NEXT BIT
3631 015116 103403 BCS 17$
3632 015120 005037 003254 CLR PR.BIT
3633 015124 000403 BR 18$
3634
3635 015126 012737 000001 003254 17$: MOV #1,PR.BIT

```

3636	015134	004737	035422	18\$:	JSR	PC,RDBIT			:SIMULATE NEXT BIT
3637	015140	005300			DEC	RO			:CHECK IF READY FOR NEXT HEADER WORD
3638	015142	001361			BNE	15\$			:NO, CONTINUE
3639	015144	005301			DEC	R1			:CHECK IF FINISHED WITH HEAD4
3640	015146	001354			BNE	12\$			:NO, CONTINUE
3641	015150	012700	000100		MOV	#64.,RO			:LOAD COUNT FOR GAP
3642	015154	013737	003254	003256	25\$:	MOV	PR.BIT,M1.BIT		:SIMULATE GAP
3643	015162	005037	003254		CLR	PR.BIT			
3644	015166	004737	035422		JSR	PC,RDBIT			
3645	015172	005300			DEC	RO			:CHECK IF GAP FINISHED
3646	015174	001367			BNE	25\$			:NO, CONTINUE
3647	015176	016237	000000	003140	MOV	RKCS1(R2),T.CS1			:GET CS1
3648	015204	016237	000010	003150	MOV	RKCS2(R2),T.CS2			:GET CS2
3649	015212	016237	000014	003154	MOV	RKER(R2),T.ER			:GET ERROR REG
3650	015220	012737	000023	003200	MOV	#WRDATA,E.CS1			:LOAD EXPECTED CS1
3651	015226	012737	000300	003210	MOV	#IR!OR,E.CS2			:LOAD EXPECTED CS2
3652	015234	012737	000400	003214	MOV	#HVRC,E.ER			:LOAD EXPECTED ERROR REG.
3653	015242	023737	003200	003140	CMP	E.CS1,T.CS1			:CHECK CS1 CORRECT
3654	015250	001401			BEQ	30\$			:YES, CHECK CS2
3655	015252	104066			ERROR	66			:CS1 INCORRECT
3656	015254	023737	003210	003150	30\$:	CMP	E.CS2,T.CS2		:CHECK CS2 CORRECT
3657	015262	001401			BEQ	32\$			:YES, CHECK ERROR REG.
3658	015264	104067			ERROR	67			:CS2 INCORRECT
3659	015266	023737	003214	003154	32\$:	CMP	E.ER,T.ER		:CHECK ERROR REG CORRECT
3660	015274	001401			BEQ	TST26			::YES, GO ON TO NEXT TEST
3661	015276	104070			ERROR	70			:ERROR REG INCORRECT

3662  
3663  
3664  
3665  
3666  
3667  
3668  
3669  
3670  
3671  
3672  
3673  
3674  
3675  
3676  
3677  
3678  
3679  
3680  
3681  
3682  
3683  
3684  
3685  
3686  
3687  
3688  
3689  
3690  
3691

```
*****  
: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  
: 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 FOLLOWING  
: 3 WORDS:  
:  
:           000100  
:           140001  
:           140101  
:  
: MAKE SURE WRITE GATE SETS INDICATING THAT HEADER HAS  
: BEEN RECOGNIZED.  
:*****  
TST26:  SCOPE  
MOV      #10.,$TIMES      ;;DO 10. ITERATIONS  
MOV      $BASE,R2        :LOAD RK611 BASE
```

```

3692 015314 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
3693 015322 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3694 015330 012762 053606 000004      MOV      #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
3695 015336 012762 177777 000002      MOV      #-1,RKWC(R2) ;WORD COUNT -1
3696 015344 012762 000001 000006      MOV      #1,RKDA(R2) ;LOAD TRACK AND SECTOR
3697 015352 012762 000100 000020      MOV      #100,RKDCYL(R2) ;LOAD CYLINDER
3698 015360 012762 000023 000000      MOV      #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3699 015366 012700 000426      MOV      #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
3700                                     ; FOR SECTOR PULSE
3701 015372 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2)
3702 015400 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3703 015406 005300      DEC      R0
3704 015410 001370      BNE     1$
3705 015412 012705 000002      MOV      #2,R5 ;LOAD HEADER COUNT
3706 015416 012703 052004      MOV      #HEAD5,R3 ;LOAD ADDRESS OF HEADERS
3707 015422 012737 000023 003200      MOV      #WRDATA,E.CS1 ;LOAD EXPECTED CS1
3708 015430 012737 000300 003210      MOV      #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3709 015436 005037 003214      CLR      E.ER ;LOAD EXPECTED MR1
3710 015442 012737 022040 003224      MOV      #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1
3711 015450 005037 003310      CLR      HDRCNT ;INITIALIZE HEADER COUNT
3712 015454 012762 000140 000026 5$:      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3713 015462 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3714 015470 005037 003254      CLR      PR.BIT ;GENERATE SYNCH
3715 015474 005037 003256      CLR      M1.BIT
3716 015500 012700 000377      MOV      #255.,R0
3717 015504 004737 035422 10$:      JSR     PC,RDBIT
3718 015510 005300      DEC      R0 ;CHECK IF SYNCH FINISHED
3719 015512 001374      BNE     10$
3720 015514 012737 000001 003254      MOV      #1,PR.BIT ;SIMULATE SYNCH BIT
3721 015522 004737 035422      JSR     PC,RDBIT
3722 015526 012701 000003      MOV      #3,R1 ;SIMULATE HEADER
3723 015532 012304 000000 12$:      MOV      (R3)+,R4 ;GET NEXT HEADER WORD
3724 015534 012700 000020      MOV      #16.,R0 ;LOAD BITS PER WORD
3725 015540 013737 003254 003256 15$:      MOV      PR.BIT,M1.BIT ;STORE PREVIOUS BIT
3726 015546 006004      ROR     R4 ;GET NEXT BIT
3727 015550 103403      BCS     17$
3728 015552 005037 003254      CLR      PR.BIT
3729 015556 000403      BR     18$
3730
3731 015560 012737 000001 003254 17$:      MOV      #1,PR.BIT
3732 015566 004737 035422 18$:      JSR     PC,RDBIT ;SIMULATE NEXT BIT
3733 015572 005300      DEC      R0 ;CHECK IF READY FOR NEXT HEADER WORD
3734 015574 001361      BNE     15$ ;NO, CONTINUE
3735 015576 005301      DEC      R1 ;CHECK IF FINISHED WITH HEADER
3736 015600 001354      BNE     12$ ;NO, CONTINUE
3737 015602 012700 000102      MOV      #66.,R0 ;LOAD COUNT FOR GAP
3738                                     ; PLUS 2 COUNTS FOR WRTGAT TO SET
3739 015606 013737 003254 003256 25$:      MOV      PR.BIT,M1.BIT ;SIMULATE GAP
3740 015614 005037 003254      CLR      PR.BIT
3741 015620 004737 035422      JSR     PC,RDBIT
3742 015624 005300      DEC      R0 ;CHECK IF GAP IS FINISHED
3743 015626 001367      BNE     25$ ;NO, CONTINUE
3744 015630 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;GET CS1
3745 015636 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;GET CS2
3746 015644 016237 000014 003154      MOV      RKER(R2),T.ER ;GET ERROR REG
3747 015652 023737 003200 003140      CMP     E.CS1,T.CS1 ;CHECK CS1 CORRECT
  
```



```
3804 016044 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
3805 016052 012762 000040 000026 MOV #DMD,RKMR1(R2)
3806 016060 005300 DEC R0
3807 016062 001370 BNE 1$
3808 016064 012705 000002 MOV #2,R5 ;LOAD HEADER COUNT
3809 016070 012703 052020 MOV #HEAD6,R3 ;LOAD ADDRESS OF HEADERS
3810 016074 012737 000023 003200 MOV #WRDATA,E.CS1 ;LOAD EXPECTED CS1
3811 016102 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3812 016110 005037 003214 CLR E.ER ;LOAD EXPECTED MR1
3813 016114 012737 022040 003224 MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1
3814 016122 005037 003310 CLR HDRCNT ;INITIALIZE HEADER COUNT
3815 016126 012762 000140 000026 5$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3816 016134 012762 000040 000026 MOV #DMD,RKMR1(R2)
3817 016142 005037 003254 CLR PR.BIT ;GENERATE SYNCH
3818 016146 005037 003256 CLR M1.BIT
3819 016152 012700 000377 MOV #255.,R0
3820 016156 004737 035422 10$: JSR PC,RDBIT
3821 016162 005300 DEC R0 ;CHECK IF SYNCH FINISHED
3822 016164 001374 BNE 10$
3823 016166 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
3824 016174 004737 035422 JSR PC,RDBIT
3825 016200 012701 000003 MOV #3,R1 ;SIMULATE HEADER
3826 016204 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
3827 016206 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
3828 016212 013737 003254 003256 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
3829 016220 006004 ROR R4 ;GET NEXT BIT
3830 016222 103403 BCS 17$
3831 016224 005037 003254 CLR PR.BIT
3832 016230 000403 BR 18$
3833
3834 016232 012737 000001 003254 17$: MOV #1,PR.BIT
3835 016240 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
3836 016244 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
3837 016246 001361 BNE 15$ ;NO, CONTINUE
3838 016250 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
3839 016252 001354 BNE 12$ ;NO, CONTINUE
3840 016254 012700 000102 MOV #66.,R0 ;LOAD COUNT FOR GAP
3841 ; PLUS 2 COUNTS FOR WRTGAT TO SET
3842 016260 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
3843 016266 005037 003254 CLR PR.BIT
3844 016272 004737 035422 JSR PC,RDBIT
3845 016276 005300 DEC R0 ;CHECK IF GAP IS FINISHED
3846 016300 001367 BNE 25$ ;NO, CONTINUE
3847 016302 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
3848 016310 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
3849 016316 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
3850 016324 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
3851 016332 001401 BEQ 30$ ;YES, CONTINUE
3852 016334 104120 ERROR 120 ;CS1 INCORRECT
3853 016336 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
3854 016344 001401 BEQ 31$ ;YES, CONTINUE
3855 016346 104121 ERROR 121 ;CS2 INCORRECT
3856 016350 023737 003214 003154 31$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
3857 016356 001401 BEQ 32$ ;YES, CONTINUE
3858 016360 104122 ERROR 122 ;ERROR REG INCORRECT
3859 016362 016237 000026 003164 32$: MOV RKMR1(R2),T.MR1 ;GET MR1
```



```
3860 016370 023737 003224 003164    CMP    E.MR1,T.MR1    ;CHECK WRITE GATE CORRECT
3861 016376 001401                BEQ    34$           ;YES, CONTINUE
3862 016400 104123                ERROR  123          ;MR1 INCORRECT
3863 016402 005237 003310    34$:  INC    HDRCNT    ;INCREMENT HEADER COUNT
3864 016406 012737 062040 003224  MOV    #WRTGAT!DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1
3865 016414 005305                DEC    R5           ;CHECK IF FINISHED
3866 016416 001402                BEQ    TST30        ;:YES, GO ON TO NEXT TEST
3867 016420 000137 016126    JMP    5$
```

\*\*\*\*\*  
\*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.

\*\*\*\*\*  
TST30: SCOPE

```
3896 016424 000004                TST30: MOV    #10, $TIMES    ;;DO 10. ITERATIONS
3897 016426 012737 000012 001200  MOV    $BASE,R2      ;LOAD RK611 BASE
3898 016434 013702 001270                MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
3899 016440 012762 100000 000000  MOV    #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3900 016446 012762 000040 000026  MOV    #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
3901 016454 012762 053606 000004  MOV    #-1,RKWC(R2)   ;WORD COUNT -1
3902 016462 012762 177777 000002  MOV    #1,RKDA(R2)   ;LOAD TRACK AND SECTOR
3903 016470 012762 000001 000006  MOV    #400,RKDCYL(R2) ;LOAD CYLINDER
3904 016476 012762 000400 000020  MOV    #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
3905 016504 012762 000023 000000  MOV    #69.*4+2,R0   ;ISSUE ENOUGH CLOCKS UNTIL READY
3906 016512 012700 000426                ; FOR SECTOR PULSE
3907                                ;
3908 016516 012762 000440 000026 1$:  MOV    #DMD!MCLK,RKMR1(R2)
3909 016524 012762 000040 000026  MOV    #DMD,RKMR1(R2)
3910 016532 005300                DEC    R0
3911 016534 001370                BNE   1$
3912 016536 012705 000002                MOV    #2,R5        ;LOAD HEADER COUNT
3913 016542 012703 052034                MOV    #HEAD7,R3    ;LOAD ADDRESS OF HEADERS
3914 016546 012737 000023 003200  MOV    #WRDATA,E.CS1 ;LOAD EXPECTED CS1
3915 016554 012737 000300 003210  MOV    #IR!OR,E.CS2 ;LOAD EXPECTED CS2
```

```
3916 016562 005037 003214          CLR      E.ER          ;LOAD EXPECTED MR1
3917 016566 012737 022040 003224  MOV      #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1
3918 016574 005037 003310          CLR      HDRCNT       ;INITIALIZE HEADER COUNT
3919 016600 012762 000140 000026 5$:  MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
3920 016606 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3921 016614 005037 003254          CLR      PR.BIT       ;GENERATE SYNCH
3922 016620 005037 003256          CLR      M1.BIT
3923 016624 012700 000377          MOV      #255.,R0
3924 016630 004737 035422          JSR      PC,RDBIT     10$:
3925 016634 005300          DEC      R0           ;CHECK IF SYNCH FINISHED
3926 016636 001374          BNE     10$
3927 016640 012737 000001 003254  MOV      #1,PR.BIT    ;SIMULATE SYNCH BIT
3928 016646 004737 035422          JSR      PC,RDBIT
3929 016652 012701 000003          MOV      #3,R1       ;SIMULATE HEADER
3930 016656 012304          MOV      (R3)+,R4    ;GET NEXT HEADER WORD
3931 016660 012700 000020          MOV      #16.,R0    ;LOAD BITS PER WORD
3932 016664 013737 003254 003256 15$:  MOV      PR.BIT,M1.BIT ;STORE PREVIOUS BIT
3933 016672 006004          ROR     R4           ;GET NEXT BIT
3934 016674 103403          BCS     17$
3935 016676 005037 003254          CLR      PR.BIT
3936 016702 000403          BR     18$
3937
3938 016704 012737 000001 003254 17$:  MOV      #1,PR.BIT
3939 016712 004737 035422          JSR      PC,RDBIT    18$:
3940 016716 005300          DEC      R0           ;SIMULATE NEXT BIT
3941 016720 001361          BNE     15$          ;CHECK IF READY FOR NEXT HEADER WORD
3942 016722 005301          DEC      R1           ;NO, CONTINUE
3943 016724 001354          BNE     12$          ;CHECK IF FINISHED WITH HEADER
3944 016726 012700 000102          MOV      #66.,R0    ;LOAD COUNT FOR GAP
3945
3946 016732 013737 003254 003256 25$:  MOV      PR.BIT,M1.BIT ; PLUS 2 COUNTS FOR WRTGAT TO SET
3947 016740 005037 003254          CLR      PR.BIT     ;SIMULATE GAP
3948 016744 004737 035422          JSR      PC,RDBIT
3949 016750 005300          DEC      R0           ;CHECK IF GAP IS FINISHED
3950 016752 001367          BNE     25$          ;NO, CONTINUE
3951 016754 016237 000000 003140  MOV      RKCS1(R2),T.CS1 ;GET CS1
3952 016762 016237 000010 003150  MOV      RKCS2(R2),T.CS2 ;GET CS2
3953 016770 016237 000014 003154  MOV      RKER(R2),T.ER  ;GET ERROR REG
3954 016776 023737 003200 003140  CMP      E.CS1,T.CS1  ;CHECK CS1 CORRECT
3955 017004 001401          BEQ     30$          ;YES, CONTINUE
3956 017006 104124          ERROR  124          ;CS1 INCORRECT
3957 017010 023737 003210 003150 30$:  CMP      E.CS2,T.CS2  ;CHECK CS2 CORRECT
3958 017016 001401          BEQ     31$          ;YES, CONTINUE
3959 017020 104125          ERROR  125          ;CS2 INCORRECT
3960 017022 023737 003214 003154 31$:  CMP      E.ER,T.ER   ;CHECK ERROR REG CORRECT
3961 017030 001401          BEQ     32$          ;YES, CONTINUE
3962 017032 104126          ERROR  126          ;ERROR REG INCORRECT
3963 017034 016237 000026 003164 32$:  MOV      RKMR1(R2),T.MR1 ;GET MR1
3964 017042 023737 003224 003164  CMP      E.MR1,T.MR1  ;CHECK WRITE GATE CORRECT
3965 017050 001401          BEQ     34$          ;YES, CONTINUE
3966 017052 104127          ERROR  127          ;MR1 INCORRECT
3967 017054 005237 003310          INC     HDRCNT       ;INCREMENT HEADER COUNT
3968 017060 012737 000200 003214  MOV      #BSE,E.ER   ;LOAD EXPECTER ERROR REG
3969 017066 005305          DEC     R5           ;CHECK IF FINISHED
3970 017070 001402          BEQ     TST31       ;:YES, GO ON TO NEXT TEST
3971 017072 000137 016600          JMP     5$
```

3972  
3973  
3974  
3975  
3976  
3977  
3978  
3979  
3980  
3981  
3982  
3983  
3984  
3985  
3986  
3987  
3988  
3989  
3990  
3991  
3992  
3993  
3994  
3995  
3996  
3997  
3998  
3999

```

*****
*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.

```

```

4000 017076 000004
4001 017100 012737 000012 001200
4002 017106 013702 001270
4003 017112 012762 100000 000000
4004 017120 012762 000040 000026
4005 017126 012762 053606 000004
4006 017134 012762 177777 000002
4007 017142 012762 000001 000006
4008 017150 012762 000140 000020
4009 017156 012762 000023 000000
4010 017164 012700 000426
4011
4012 017170 012762 000440 000026 1$:
4013 017176 012762 000040 000026
4014 017204 005300
4015 017206 001370
4016 017210 012705 000002
4017 017214 012703 052050
4018 017220 012737 000023 003200
4019 017226 012737 000300 003210
4020 017234 005037 003214
4021 017240 012737 022040 003224
4022 017246 005037 003310
4023 017252 012762 000140 000026 5$:
4024 017260 012762 000040 000026
4025 017266 005037 003254
4026 017272 005037 003256
4027 017276 012700 000377

```

```

*****
TST31: SCOPE
MOV #10.,$TIMES ;;DO 10. ITERATIONS
MOV $BASE,R2 ;;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
MOV #-1,RKWC(R2) ;WORD COUNT -1
MOV #1,RKDA(R2) ;LOAD TRACK AND SECTOR
MOV #140,RKDCYL(R2) ;LOAD CYLINDER
MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
MOV #69.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
; FOR SECTOR PULSE
1$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV #2,R5 ;LOAD HEADER COUNT
MOV #HEAD8,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
5$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
MOV #DMD,RKMR1(R2)
CLR PR.BIT ;GENERATE SYNCH
CLR M1.BIT
MOV #255.,R0

```

```
4028 017302 004737 035422 10$: JSR PC,RDBIT
4029 017306 005300 DEC RO ;CHECK IF SYNCH FINISHED
4030 017310 001374 BNE 10$
4031 017312 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
4032 017320 004737 035422 JSR PC,RDBIT
4033 017324 012701 000003 MOV #3,R1 ;SIMULATE HEADER
4034 017330 012304 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
4035 017332 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
4036 017336 013737 003254 003256 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4037 017344 006004 ROR R4 ;GET NEXT BIT
4038 017346 103403 BCS 17$
4039 017350 005037 003254 CLR PR.BIT
4040 017354 000403 BR 18$
4041
4042 017356 012737 000001 003254 17$: MOV #1,PR.BIT
4043 017364 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4044 017370 005300 DEC RO ;CHECK IF READY FOR NEXT HEADER WORD
4045 017372 001361 BNE 15$ ;NO, CONTINUE
4046 017374 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4047 017376 001354 BNE 12$ ;NO, CONTINUE
4048 017400 012700 000102 MOV #66.,R0 ;LOAD COUNT FOR GAP
4049 ; PLUS 2 COUNTS FOR WRTGAT TO SET
4050 017404 013737 003254 003256 25$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
4051 017412 005037 003254 CLR PR.BIT
4052 017416 004737 035422 JSR PC,RDBIT
4053 017422 005300 DEC RO ;CHECK IF GAP IS FINISHED
4054 017424 001367 BNE 25$ ;NO, CONTINUE
4055 017426 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4056 017434 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4057 017442 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
4058 017450 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4059 017456 001401 BEQ 30$ ;YES, CONTINUE
4060 017460 104130 ERROR 130 ;CS1 INCORRECT
4061 017462 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4062 017470 001401 BEQ 31$ ;YES, CONTINUE
4063 017472 104131 ERROR 131 ;CS2 INCORRECT
4064 017474 023737 003214 003154 31$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4065 017502 001401 BEQ 32$ ;YES, CONTINUE
4066 017504 104132 ERROR 132 ;ERROR REG INCORRECT
4067 017506 016237 000026 003164 32$: MOV RMR1(R2),T.MR1 ;GET MR1
4068 017514 023737 003224 003164 CMP E.MR1,T.MR1 ;CHECK WRITE GATE CORRECT
4069 017522 001401 BEQ 34$ ;YES, CONTINUE
4070 017524 104133 ERROR 133 ;MR1 INCORRECT
4071 017526 005237 003310 34$: INC HDRCNT ;INCREMENT HEADER COUNT
4072 017532 012737 000400 003214 MOV #HVRC,E.ER ;LOAD EXPECTER ERROR REG
4073 017540 005305 DEC R5 ;CHECK IF FINISHED
4074 017542 001402 BEQ TST32 ;:YES, GO ON TO NEXT TEST
4075 017544 000137 017252 JMP 5$
```

```
4076
4077 ;*****
4078 ;*TEST 32 OPI AND HVRC ON LAST HEADER
4079 ;*
4080 ;* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4081 ;* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
4082 ;* DATA OF ONE WORD TO AN RK06 IN 26 SECTOR FORMAT,
4083 ;* CYLINDER 240, HEAD 0, SECTOR 1. CLOCK THROUGH
```

```

4084 : * SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
4085 : * AND 30 HEADERS CONSISTING OF THE FOLLOWING 3 WORDS:
4086 : *
4087 : *      000240
4088 : *      140000
4089 : *      140240
4090 : *
4091 : * MAKE SURE NO ERROR IS REPORTED AND WRITE GATE
4092 : * DOES NOT SET. SIMULATE A SECTOR PULSE AND A
4093 : * HEADER CONSISTING OF THE FOLLOWING THREE WORDS:
4094 : *
4095 : *      000240
4096 : *      140000
4097 : *      140040
4098 : *
4099 : * MAKE SURE HEADER VRC AND OPI ERROR SET.
4100 : *
4101 : *

```

```

4102 017550 000004 TST32: SCOPE
4103 017552 012737 000012 001200 MOV #10.,$TIMES ;;DO 10. ITERATIONS
4104 017560 013702 001270 MOV $BASE,R2 ;:LOAD RK611 BASE
4105 017564 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
4106 017572 012700 002500 MOV #2500,R0 ;:SET COUNT FOR STALL
4107 017576 005300 2$: DEC R0 ;:DEC COUNT
4108 017600 001376 BNE 2$ ;:LOOP UNTIL 0
4109 017602 012762 000040 000026 MOV #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
4110 017610 012762 053606 000004 MOV #BUFF,RKBA(R2) ;:LOAD DUMMY BUFFER ADDRESS
4111 017616 012762 177777 000002 MOV #-1,RKWC(R2) ;:WORD COUNT = 1
4112 017624 012762 000240 000020 MOV #240,RKDCYL(R2) ;:LOAD CYLINDER
4113 017632 012762 000001 000006 MOV #1,RKDA(R2) ;:LOAD TRACK AND SECTOR
4114 017640 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;:ISSUE WRITE DATA
4115 017646 012700 000426 MOV #69.*4+2,R0 ;:ISSUE ENOUGH CLODKS UNTIL READY
4116 : ;: FOR SECTOR PULSE
4117 017652 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4118 017660 012762 000040 000026 MOV #DMD,RKMR1(R2)
4119 017666 005300 DEC R0
4120 017670 001370 BNE 1$
4121 017672 012705 000040 MOV #32.,R5 ;:LOAD HEADER COUNT
4122 017676 012703 052464 MOV #OPI2,R3 ;:LOAD ADDRESS OF HEADERS
4123 017702 012737 000023 003200 MOV #WRDATA,E.CS1 ;:LOAD EXPECTED CS1
4124 017710 012737 000300 003210 MOV #IR!OR,E.CS2 ;:LOAD EXPECTED CS2
4125 017716 005037 003214 CLR E.ER ;:LOAD EXPECTED ERROR REG
4126 017722 012737 022040 003224 MOV #DMD!MEWD!ECCW,E.MR1 ;:LOAD EXPECTED MR1
4127 017730 005037 003310 CLR HDRCNT ;:INITIALIZE HEADER COUNT
4128 017734 012762 000140 000026 5$: MOV #DMD!MSP,RKMR1(R2) ;:SIMULATE SECTOR PULSE
4129 017742 012762 000040 000026 MOV #DMD,RKMR1(R2)
4130 017750 022737 000037 003310 CMP #31.,HDRCNT ;:CHECK IF ALL HEADERS DONE
4131 017756 001460 BEQ 26$ ;:YES - SKIP TO ERROR TEST
4132 017760 005037 003254 CLR PR.BIT ;:GENERATE SYNCH
4133 017764 005037 003256 CLR M1.BIT
4134 017770 012700 000377 MOV #255.,R0
4135 017774 004737 035422 10$: JSR PC,RDBIT
4136 020000 005300 DEC R0 ;:CHECK IF SYNCH FINISHED
4137 020002 001374 BNE 10$
4138 020004 012737 000001 003254 MOV #1,PR.BIT ;:SIMULATE SYNCH BIT
4139 020012 004737 035422 JSR PC,RDBIT

```

```
4140 020016 012701 000003      MOV      #3,R1      ;SIMULATE OPI
4141 020022 012304      12$: MOV      (R3)+,R4    ;GET NEXT HEADER WORD
4142 020024 012700 000020      MOV      #16.,R0    ;LOAD BITS PER WORD
4143 020030 013737 003254 003256 15$: MOV      PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4144 020036 006004      ROR      R4         ;GET NEXT BIT
4145 020040 103403      BCS      17$
4146 020042 005037 003254      CLR      PR.BIT
4147 020046 000403      BR       18$
4148
4149 020050 012737 000001 003254 17$: MOV      #1,PR.BIT
4150 020056 004737 035422 18$: JSR      PC,RDBIT    ;SIMULATE NEXT BIT
4151 020062 005300      DEC      R0         ;CHECK IF READY FOR NEXT HEADER WORD
4152 020064 001361      BNE      15$        ;NO, CONTINUE
4153 020066 005301      DEC      R1         ;CHECK IF FINISHED WITH HEADER
4154 020070 001354      BNE      12$        ;NO,CONTINUE
4155 020072 012700 000100      MOV      #64.,R0    ;LOAD COUNT FOR GAP
4156 020076 013737 003254 003256 25$: MOV      PR.BIT,M1.BIT ;SIMULATE GAP
4157 020104 005037 003254      CLR      PR.BIT
4158 020110 004737 035422      JSR      PC,RDBIT
4159 020114 005300      DEC      R0         ;CHECK IF GAP IS FINISHED
4160 020116 001367      BNE      25$        ;NO, CONTINUE
4161 020120 016237 000000 003140 26$: MOV      RKCS1(R2),T.CS1 ;GET CS1
4162 020126 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;GET CS2
4163 020134 016237 000014 003154      MOV      RKER(R2),T.ER  ;GET ERROR REG.
4164 020142 023737 003200 003140      CMP      E.CS1,T.CS1   ;CHECK CS1 CORRECT
4165 020150 001401      BEQ      30$        ;YES, CONTINUE
4166 020152 104100      ERROR   100        ;CS1 INCORRECT
4167 020154 023737 003210 003150 30$: CMP      E.CS2,T.CS2   ;CHECK CS2 CORRECT
4168 020162 001401      BEQ      31$        ;YES, CONTINUE
4169 020164 104101      ERROR   101        ;CS2 INCORRECT
4170 020166 023737 003214 003154 31$: CMP      E.ER,T.ER    ;CHECK ERROR REG CORRECT
4171 020174 001401      BEQ      32$        ;YES, CONTINUE
4172 020176 104102      ERROR   102        ;ERROR REG INCORRECT
4173 020200 016237 000026 003164 32$: MOV      RMR1(R2),T.MR1 ;GET MR1
4174 020206 023737 003224 003164      CMP      E.MR1,T.MR1   ;CHECK TO MAKE SURE WRITE GATE DID NOT SET
4175 020214 001401      BEQ      34$        ;YES, CONTINUE
4176 020216 104103      ERROR   103        ;MR1 INCORRECT
4177 020220 005237 003310      INC      HDRCNT      ;INCREMENT HEADER COUNT
4178 020224 022737 000037 003310      CMP      #31.,HDRCNT  ;CHECK IF LAST HEADER
4179 020232 001011      BNE      35$        ;NO, CHECK IF FINSHED
4180 020234 012737 020400 003214      MOV      #OPI!HVRC,E.ER ;LOAD ERROR BIT
4181 020242 042737 000001 003200      BIC      #GO,E.CS1    ;ADJUST E.CS1 FOR END OF OP CONTENTS
4182 020250 052737 100200 003200      BIS      #RDY!CERR,E.CS1
4183 020256 005305      35$: DEC      R5         ;CHECK IF FINISHED
4184 020260 001402      BEQ      37$        ;YES - SKIP
4185 020262 000137 017734      JMP      5$         ;DO NEXT SECTOR
4186 020266 012762 100000 000000 37$: MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4187 020274 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;GET CS1
4188 020302 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;CS2
4189 020310 016237 000014 003154      MOV      RKER(R2),T.ER  ;ER
4190 020316 012737 000200 003200      MOV      #RDY,E.CS1    ;SET EXPECTED CS1
4191 020324 023737 003140 003200      CMP      T.CS1,E.CS1   ;CHECK IF CORRECT
4192 020332 001401      BEQ      TST33      ;GO TO NEXT TEST
4193 020334 104153      ERROR   153
4194
4195
```

;;\*\*\*\*\*

4196  
4197  
4198  
4199  
4200  
4201  
4202  
4203  
4204  
4205  
4206  
4207  
4208  
4209  
4210  
4211  
4212  
4213  
4214  
4215  
4216  
4217  
4218  
4219  
4220  
4221  
4222  
4223  
4224  
4225  
4226  
4227  
4228 020336 000004  
4229 020340 012737 000012 001200  
4230 020346 013702 001270  
4231 020352 012762 100000 000000  
4232 020360 012700 002500  
4233 020364 005300  
4234 020366 001376  
4235 020370 012762 000040 000026  
4236 020376 012762 053606 000004  
4237 020404 012762 177777 000002  
4238 020412 012762 000300 000020  
4239 020420 012762 000001 000006  
4240 020426 012762 000023 000000  
4241 020434 012700 000426  
4242  
4243 020440 012762 000440 000026  
4244 020446 012762 000040 000026  
4245 020454 005300  
4246 020456 001370  
4247 020460 012705 000040  
4248 020464 012703 052764  
4249 020470 012737 000023 003200  
4250 020476 012737 000300 003210  
4251 020504 005037 003214

```
;*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.
;*****
TST33: SCOPE
MOV #10,$TIMES ;;DO 10. ITERATIONS
MOV $BASE,R2 ;;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;;CLEAR RK611
MOV #2500,R0 ;;SET COUNT FOR STALL
2$: DEC R0 ;;DEC COUNT
BNE 2$ ;;LOOP UNTIL 0
MOV #DMD,RKMR1(R2) ;;PUT RK611 IN DIAGNOSTIC MODE
MOV #BUFF,RKBA(R2) ;;LOAD DUMMY BUFFER ADDRESS
MOV #-1,RKWC(R2) ;;WORD COUNT = 1
MOV #300,RKDCYL(R2) ;;LOAD CYLINDER
MOV #1,RKDA(R2) ;;LOAD TRACK AND SECTOR
MOV #WRDATA,RKCS1(R2) ;;ISSUE WRITE DATA
MOV #69.*4+2,R0 ;;ISSUE ENOUGH CLODKS UNTIL READY
; FOR SECTOR PULSE
1$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV #32,R5 ;;LOAD HEADER COUNT
MOV #OPI3,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 ERROR REG
```

```

4252 020510 012737 022040 003224      MOV      #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MR1
4253 020516 005037 003310              CLR      HDRCNT                ;INITIALIZE HEADER COUNT
4254 020522 012762 000140 000026 5$:    MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4255 020530 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4256 020536 022737 000037 003310      CMP      #31.,HDRCNT          ;CHECK IF ALL HEADERS DONE
4257 020544 001460              BEQ      26$                  ;YES - SKIP TO ERROR TEST
4258 020546 005037 003254              CLR      PR.BIT              ;GENERATE SYNCH
4259 020552 005037 003256              CLR      M1.BIT
4260 020556 012700 000377              MOV      #255.,R0
4261 020562 004737 035422 10$:    JSR      PC,RDBIT
4262 020566 005300              DEC      R0                  ;CHECK IF SYNCH FINISHED
4263 020570 001374              BNE      10$
4264 020572 012737 000001 003254      MOV      #1,PR.BIT          ;SIMULATE SYNCH BIT
4265 020600 004737 035422              JSR      PC,RDBIT
4266 020604 012701 000003              MOV      #3,R1              ;SIMULATE OPI
4267 020610 012304 12$:    MOV      (R3)+,R4            ;GET NEXT HEADER WORD
4268 020612 012700 000020              MOV      #16.,R0           ;LOAD BITS PER WORD
4269 020616 013737 003254 003256 15$:    MOV      PR.BIT,M1.BIT      ;STORE PREVIOUS BIT
4270 020624 006004              ROR      R4                  ;GET NEXT BIT
4271 020626 103403              BCS      17$
4272 020630 005037 003254              CLR      PR.BIT
4273 020634 000403              BR       18$
4274
4275 020636 012737 000001 003254 17$:    MOV      #1,PR.BIT
4276 020644 004737 035422 18$:    JSR      PC,RDBIT          ;SIMULATE NEXT BIT
4277 020650 005300              DEC      R0                  ;CHECK IF READY FOR NEXT HEADER WORD
4278 020652 001361              BNE      15$                ;NO, CONTINUE
4279 020654 005301              DEC      R1                  ;CHECK IF FINISHED WITH HEADER
4280 020656 001354              BNE      12$                ;NO,CONTINUE
4281 020660 012700 000100              MOV      #64.,R0           ;LOAD COUNT FOR GAP
4282 020664 013737 003254 003256 25$:    MOV      PR.BIT,M1.BIT      ;SIMULATE GAP
4283 020672 005037 003254              CLR      PR.BIT
4284 020676 004737 035422              JSR      PC,RDBIT
4285 020702 005300              DEC      R0                  ;CHECK IF GAP IS FINISHED
4286 020704 001367              BNE      25$                ;NO, CONTINUE
4287 020706 016237 000000 003140 26$:    MOV      RKCS1(R2),T.CS1    ;GET CS1
4288 020714 016237 000010 003150      MOV      RKCS2(R2),T.CS2    ;GET CS2
4289 020722 016237 000014 003154      MOV      RKER(R2),T.ER      ;GET ERROR REG.
4290 020730 023737 003200 003140      CMP      E.CS1,T.CS1        ;CHECK CS1 CORRECT
4291 020736 001401              BEQ      30$                ;YES, CONTINUE
4292 020740 104104              ERROR    104                ;CS1 INCORRECT
4293 020742 023737 003210 003150 30$:    CMP      E.CS2,T.CS2        ;CHECK CS2 CORRECT
4294 020750 001401              BEQ      31$                ;YES, CONTINUE
4295 020752 104105              ERROR    105                ;CS2 INCORRECT
4296 020754 023737 003214 003154 31$:    CMP      E.ER,T.ER          ;CHECK ERROR REG CORRECT
4297 020762 001401              BEQ      32$                ;YES, CONTINUE
4298 020764 104106              ERROR    106                ;ERROR REG INCORRECT
4299 020766 016237 000026 003164 32$:    MOV      RKMR1(R2),T.MR1    ;GET MR1
4300 020774 023737 003224 003164      CMP      E.MR1,T.MR1        ;CHECK TO MAKE SURE WRITE GATE DID NOT SET
4301 021002 001401              BEQ      34$                ;YES, CONTINUE
4302 021004 104107              ERROR    107                ;MR1 INCORRECT
4303 021006 005237 003310 34$:    INC      HDRCNT            ;INCREMENT HEADER COUNT
4304 021012 022737 000037 003310      CMP      #31.,HDRCNT        ;CHECK IF LAST HEADER
4305 021020 001011              BNE      35$                ;NO, CHECK IF FINSHED
4306 021022 012737 020400 003214      MOV      #OPI!HVRC,E.ER     ;LOAD ERROR BIT
4307 021030 042737 000001 003200      BIC      #GO,E.CS1          ;ADJUST E.CS1 FOR END OF OP CONTENTS

```



4308	021036	052737	100200	003200		BIS	#RDY!CERR,E.CS1	
4309	021044	005305			35\$:	DEC	R5	;CHECK IF FINISHED
4310	021046	001402				BEQ	37\$	;YES - SKIP
4311	021050	000137	020522			JMP	5\$	;DO NEXT SECTOR
4312	021054	012762	100000	000000	37\$:	MOV	#CCLR,RKCS1(R2)	;CLEAR CONTROLLER
4313	021062	016237	000000	003140		MOV	RKCS1(R2),T.CS1	;GET CS1
4314	021070	016237	000010	003150		MOV	RKCS2(R2),T.CS2	; CS2
4315	021076	016237	000014	003154		MOV	RKER(R2),T.ER	; ER
4316	021104	012737	000200	003200		MOV	#RDY,E.CS1	;SET EXPECTED CS1
4317	021112	023737	003140	003200		MOV	T.CS1,E.CS1	;CHECK IF CORRECT
4318	021120	001401				BEQ	TST34	;GO TO NEXT TEST
4319	021122	104153				ERROR	153	

4320  
4321  
4322  
4323  
4324  
4325  
4326  
4327  
4328  
4329  
4330  
4331  
4332  
4333  
4334  
4335  
4336  
4337  
4338  
4339  
4340  
4341  
4342  
4343  
4344  
4345  
4346

```
*****  
*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  
*  
* MAKE SURE HEADER VRC AND OPI ERRORS SET.  
*****
```

4347	021124	000004			TST34:	SCOPE		
4348	021126	012737	000012	001200		MOV	#10,\$TIMES	;DO 10. ITERATIONS
4349	021134	013702	001270			MOV	\$BASE,R2	;LOAD RK611 BASE
4350	021140	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
4351	021146	012700	002500			MOV	#2500,R0	;SET COUNT FOR STALL
4352	021152	005300			2\$:	DEC	R0	;DEC COUNT
4353	021154	001376				BNE	2\$	;LOOP UNTIL 0
4354	021156	012762	000040	000026		MOV	#DMD,RKMR1(R2)	;PUT RK611 IN DIAGNOSTIC MODE
4355	021164	012762	053606	000004		MOV	#BUFF,RKBA(R2)	;LOAD DUMMY BUFFER ADDRESS
4356	021172	012762	177777	000002		MOV	#-1,RKWC(R2)	;WORD COUNT = 1
4357	021200	012762	000040	000020		MOV	#40,RKDCYL(R2)	;LOAD CYLINDER
4358	021206	012762	000001	000006		MOV	#1,RKDA(R2)	;LOAD TRACK AND SECTOR
4359	021214	012762	000023	000000		MOV	#WRDATA,RKCS1(R2)	;ISSUE WRITE DATA
4360	021222	012700	000426			MOV	#69.*4+2,R0	;ISSUE ENOUGH CLODKS UNTIL READY ; FOR SECTOR PULSE
4361								
4362	021226	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
4363	021234	012762	000040	000026		MOV	#DMD,RKMR1(R2)	

4364	021242	005300			DEC	R0	
4365	021244	001370			BNE	1\$	
4366	021246	012705	000040		MOV	#32.,R5	;LOAD HEADER COUNT
4367	021252	012703	053264		MOV	#OPI4,R3	;LOAD ADDRESS OF HEADERS
4368	021256	012737	000023	003200	MOV	#WRDATA,E.CS1	;LOAD EXPECTED CS1
4369	021264	012737	000300	003210	MOV	#IR!OR,E.CS2	;LOAD EXPECTED CS2
4370	021272	005037	003214		CLR	E.ER	;LOAD EXPECTED ERROR REG
4371	021276	012737	022040	003224	MOV	#DMD!MEWD!ECCW,E.MR1	;LOAD EXPECTED MR1
4372	021304	005037	003310		CLR	HDRCNT	;INITIALIZE HEADER COUNT
4373	021310	012762	000140	000026	5\$: MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
4374	021316	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
4375	021324	022737	000037	003310	CMP	#31.,HDRCNT	;CHECK IF ALL HEADERS DONE
4376	021332	001460			BEQ	26\$	;YES - SKIP TO ERROR TEST
4377	021334	005037	003254		CLR	PR.BIT	;GENERATE SYNCH
4378	021340	005037	003256		CLR	M1.BIT	
4379	021344	012700	000377		MOV	#255.,R0	
4380	021350	004737	035422		10\$: JSR	PC,RDBIT	
4381	021354	005300			DEC	R0	;CHECK IF SYNCH FINISHED
4382	021356	001374			BNE	10\$	
4383	021360	012737	000001	003254	MOV	#1,PR.BIT	;SIMULATE SYNCH BIT
4384	021366	004737	035422		JSR	PC,RDBIT	
4385	021372	012701	000003		MOV	#3,R1	;SIMULATE OPI
4386	021376	012304			12\$: MOV	(R3)+,R4	;GET NEXT HEADER WORD
4387	021400	012700	000020		MOV	#16.,R0	;LOAD BITS PER WORD
4388	021404	013737	003254	003256	15\$: MOV	PR.BIT,M1.BIT	;STORE PREVIOUS BIT
4389	021412	006004			ROR	R4	;GET NEXT BIT
4390	021414	103403			BCS	17\$	
4391	021416	005037	003254		CLR	PR.BIT	
4392	021422	000403			BR	18\$	
4393							
4394	021424	012737	000001	003254	17\$: MOV	#1,PR.BIT	
4395	021432	004737	035422		18\$: JSR	PC,RDBIT	;SIMULATE NEXT BIT
4396	021436	005300			DEC	R0	;CHECK IF READY FOR NEXT HEADER WORD
4397	021440	001361			BNE	15\$	;NO, CONTINUE
4398	021442	005301			DEC	R1	;CHECK IF FINISHED WITH HEADER
4399	021444	001354			BNE	12\$	;NO,CONTINUE
4400	021446	012700	000100		MOV	#64.,R0	;LOAD COUNT FOR GAP
4401	021452	013737	003254	003256	25\$: MOV	PR.BIT,M1.BIT	;SIMULATE GAP
4402	021460	005037	003254		CLR	PR.BIT	
4403	021464	004737	035422		JSR	PC,RDBIT	
4404	021470	005300			DEC	R0	;CHECK IF GAP IS FINISHED
4405	021472	001367			BNE	25\$	;NO, CONTINUE
4406	021474	016237	000000	003140	26\$: MOV	RKCS1(R2),T.CS1	;GET CS1
4407	021502	016237	000010	003150	MOV	RKCS2(R2),T.CS2	;GET CS2
4408	021510	016237	000014	003154	MOV	RKER(R2),T.ER	;GET ERROR REG.
4409	021516	023737	003200	003140	CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
4410	021524	001401			BEQ	30\$	;YES, CONTINUE
4411	021526	104110			ERROR	110	;CS1 INCORRECT
4412	021530	023737	003210	003150	30\$: CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
4413	021536	001401			BEQ	31\$	;YES, CONTINUE
4414	021540	104111			ERROR	111	;CS2 INCORRECT
4415	021542	023737	003214	003154	31\$: CMP	E.ER,T.ER	;CHECK ERROR REG CORRECT
4416	021550	001401			BEQ	32\$	;YES, CONTINUE
4417	021552	104112			ERROR	112	;ERROR REG INCORRECT
4418	021554	016237	000026	003164	32\$: MOV	RKMR1(R2),T.MR1	;GET MR1
4419	021562	023737	003224	003164	CMP	E.MR1,T.MR1	;CHECK TO MAKE SURE WRITE GATE DID NOT SET

```

4420 021570 001401 BEQ 34$ :YES, CONTINUE
4421 021572 104113 ERROR 113 :MR1 INCORRECT
4422 021574 005237 003310 34$: INC HDRCNT :INCREMENT HEADER COUNT
4423 021600 022737 000037 003310 CMP #31.,HDRCNT :CHECK IF LAST HEADER
4424 021606 001011 BNE 35$ :NO, CHECK IF FINISHED
4425 021610 012737 020400 003214 MOV #OPI!HVRC,E.ER :LOAD ERROR BIT
4426 021616 042737 000001 003200 BIC #GO,E.CS1 :ADJUST E.CS1 FOR END OF OP CONTENTS
4427 021624 052737 100200 003200 BIS #RDY!CERR,E.CS1
4428 021632 005305 35$: DEC R5 :CHECK IF FINISHED
4429 021634 001402 BEQ 37$ :YES - SKIP
4430 021636 000137 021310 JMP 5$ :DO NEXT SECTOR
4431 021642 012762 100000 000000 37$: MOV #CCLR,RKCS1(R2) :CLEAR CONTROLLER
4432 021650 016237 000000 003140 MOV RKCS1(R2),T.CS1 :GET CS1
4433 021656 016237 000010 003150 MOV RKCS2(R2),T.CS2 : CS2
4434 021664 016237 000014 003154 MOV RKER(R2),T.ER : ER
4435 021672 012737 000200 003200 MOV #RDY,E.CS1 :SET EXPECTED CS1
4436 021700 023737 003140 003200 CMP T.CS1,E.CS1 :CHECK IF CORRECT
4437 021706 001401 BEQ TST35 ;;GO TO NEXT TEST
4438 021710 104153 ERROR 153

```

```

4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452
4453

```

```

*****
*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.
*****

```

```

4454 021712 000004 TST35: SCOPE
4455 021714 012737 000010 001200 MOV #10,$TIMES ;;DO 10 ITERATIONS
4456 021722 013702 001270 000000 MOV $BASE,R2 :LOAD RK611 BASE
4457 021726 012762 100000 000000 MOV #CCLR,RKCS1(R2) :CLEAR RK611
4458 021734 012762 000040 000026 MOV #DMD,RKMR1(R2) :PUT RK611 IN DIAGNOSTIC MODE
4459 021742 012762 053606 000004 MOV #BUFF,RKBA(R2) :LOAD DUMMY BU. ADDRESS
4460 021750 012762 177777 000002 MOV #-1,RKWC(R2) :WORD COUNT=1
4461 021756 012762 000000 000020 MOV #0,RKDCYL(R2) :LOAD CYLINDER 11DMS
4462 021764 012762 000000 000006 MOV #0,RKDA(R2) :LOAD TRACK AND SECTOR
4463 021772 012762 000023 000000 MOV #WRDATA,RKCS1(R2) :ISSUE COMMAND
4464 022000 012700 000426 MOV #69.*4+2,R0 :ISSUE ENOUGH CLOCKS UNTIL
4465 : READY FOR SECTOR PULSE.
4466 022004 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4467 022012 012762 000040 000026 MOV #DMD,RKMR1(R2)
4468 022020 005300 DEC R0
4469 022022 001370 BNE 1$
4470 022024 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4471 022032 012762 000040 000026 MOV #DMD,RKMR1(R2)
4472 022040 005037 003254 CLR PR.BIT ;GENERATE SYNCH
4473 022044 005037 003256 CLR M1.BIT
4474 022050 012700 000377 MOV #255.,R0
4475 022054 004737 035422 5$: JSR PC,RDBIT

```

```

4476 022060 005300 DEC R0 ;CHECK IF SYNCH FINISHED
4477 022062 001374 BNE 5$
4478 022064 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH
4479 022072 004737 035422 JSR PC,RDBIT
4480 022076 012703 051762 MOV #HEAD2,R3 ;LOAD HEADER
4481 022102 012701 000003 MOV #3,R1 ;LOAD WORDS PER HEADER
4482 022106 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
4483 022110 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
4484 022114 013737 003254 003256 12$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4485 022122 006004 ROR R4 ;GET NEXT BIT
4486 022124 103403 BCS 15$
4487 022126 005037 003254 CLR PR.BIT
4488 022132 000403 BR 16$
4489
4490 022134 012737 000001 003254 15$: MOV #1,PR.BIT
4491 022142 004737 035422 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4492 022146 005300 DEC R0 ;CHECK IF READY FOR NEXT WORD
4493 022150 001361 BNE 12$ ;NO, CONTINUE
4494 022152 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
4495 022154 001354 BNE 10$ ;NO, CONTINUE
4496 022156 012700 000101 MOV #65,R0
4497 022162 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
4498 022170 005037 003254 CLR PR.BIT
4499 022174 004737 035422 JSR PC,RDBIT
4500 022200 005300 DEC R0 ;CHECK IF GAP FINISHED
4501 022202 001367 BNE 20$ ;NO, CONTINUE
4502 022204 012700 021200 MOV #2*<256.+<16.*256.>+64.>,R0 ;LOAD COUNT UNTIL POSTAMBLE
4503 022210 012762 000440 000026 25$: MOV #DMD!MCLK,RKMR1(R2)
4504 022216 012762 000040 000026 MOV #DMD,RKMR1(R2)
4505 022224 005300 DEC R0
4506 022226 001370 BNE 25$
4507 022230 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4508 022236 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4509 022244 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
4510 022252 012737 100222 003200 MOV #CERR!RDY!WRDATA<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4511 022260 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4512 022266 012737 000200 003214 MOV #BSE,E.ER ;LOAD EXPECTED ERROR REG
4513 022274 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4514 022302 001401 BEQ 30$ ;YES, CONTINUE
4515 022304 104136 ERROR 136 ;CS1 INCORRECT
4516 022306 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4517 022314 001401 BEQ 32$ ;YES, CONTINUE
4518 022316 104137 ERROR 137 ;CS2 INCORRECT
4519 022320 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4520 022326 001401 BEQ 35$ ;YES - SKIP
4521 022330 104140 ERROR 140 ;ERROR REG INCORRECT
4522 022332 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4523 022340 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4524 022346 016237 000010 003150 MOV RKCS2(R2),T.CS2 ; CS2
4525 022354 016237 000014 003154 MOV RKER(R2),T.ER ; ER
4526 022362 012737 000200 003200 MOV #200,E.CS1 ;SET EXPECTED CS1
4527 022370 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4528 022376 001401 BEQ TST36 ;GO TO NEXT TEST
4529 022400 104153 ERROR 153 ;ERROR DID NOT CLEAR

```

4530  
 4531  
 ;:\*\*\*\*\*

```

4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544 022402 000004
4545 022404 012737 000012 001200
4546 022412 013702 001270
4547 022416 012762 100000 000000
4548 022424 012762 000040 000026
4549 022432 012762 053606 000004
4550 022440 012762 177777 000002
4551 022446 012762 000300 000020
4552 022454 012762 000000 000006
4553 022462 012762 000023 000000
4554 022470 012700 000426
4555
4556 022474 012762 000440 000026 1$:
4557 022502 012762 000040 000026
4558 022510 005300
4559 022512 001370
4560 022514 012762 000140 000026
4561 022522 012762 000040 000026
4562 022530 005037 003254
4563 022534 005037 003256
4564 022540 012700 000377
4565 022544 004737 035422 5$:
4566 022550 005300
4567 022552 001374
4568 022554 012737 000001 003254
4569 022562 004737 035422
4570 022566 012703 052064
4571 022572 012701 000003
4572 022576 012304 10$:
4573 022600 012700 000020
4574 022604 013737 003254 003256 12$:
4575 022612 006004
4576 022614 103403
4577 022616 005037 003254
4578 022622 000403
4579
4580 022624 012737 000001 003254 15$:
4581 022632 004737 035422 16$:
4582 022636 005300
4583 022640 001361
4584 022642 005301
4585 022644 001354
4586 022646 012700 000101
4587 022652 013737 003254 003256 20$:
    
```

```

:*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.
:*
*****
TST36: SCOPE
MOV #10, $TIMES ;;DO 10. ITERATIONS
MOV $BASE, R2 ;LOAD RK611 BASE
MOV #CCLR, RKCS1(R2) ;CLEAR RK611
MOV #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #BUFF, RKBA(R2) ;LOAD DUMMY BUS ADDRESS
MOV #-1, RKWC(R2) ;WORD COUNT=1
MOV #300, RKDCYL(R2) ;LOAD CYLINDER 11DMS
MOV #0, RKDA(R2) ;LOAD TRACK AND SECTOR
MOV #WRDATA, RKCS1(R2) ;ISSUE COMMAND
MOV #69. +4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL
; READY FOR SECTOR PULSE.
1$: 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
5$: JSR PC, RDBIT
DEC R0 ;CHECK IF SYNCH FINISHED
BNE 5$
MOV #1, PR.BIT ;SIMULATE SYNCH
JSR PC, RDBIT
MOV #HEAD10, R3 ;LOAD HEADER
MOV #3, R1 ;LOAD WORDS PER HEADER
MOV (R3)+, R4 ;GET NEXT WORD
MOV #16., R0 ;LOAD BITS PER WORD
12$: MOV PR.BIT, M1.BIT ;STORE PREVIOUS BIT
ROR R4 ;GET NEXT BIT
BCS 15$
CLR PR.BIT
BR 16$
15$: MOV #1, PR.BIT
16$: JSR PC, RDBIT ;SIMULATE NEXT BIT
DEC R0 ;CHECK IF READY FOR NEXT WORD
BNE 12$ ;NO, CONTINUE
DEC R1 ;CHECK IF FINISHED WITH HEADER
BNE 10$ ;NO, CONTINUE
MOV #65., R0
20$: MOV PR.BIT, M1.BIT ;SIMULATE GAP
    
```

```

4588 022660 005037 003254 CLR PR.BIT
4589 022664 004737 035422 JSR PC,RDBIT
4590 022670 005300 DEC R0 ;CHECK IF GAP FINISHED
4591 022672 001367 BNE 20$ ;NO, CONTINUE
4592 022674 012700 021200 MOV #2*<256.+<16.*256.>+64.>,R0 ;LOAD COUNT UNTIL POSTAMBLE
4593 022700 012762 000440 000026 25$: MOV #DMD!MCLK,RKMR1(R2)
4594 022706 012762 000040 000026 MOV #DMD,RKMR1(R2)
4595 022714 005300 DEC R0
4596 022716 001370 BNE 25$
4597 022720 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4598 022726 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;GET CS2
4599 022734 016237 000014 003154 MOV RKER(R2),T.ER ;GET ERROR REG
4600 022742 012737 100222 003200 MOV #CERR!RDY!WRDATA<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4601 022750 012737 000300 003210 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4602 022756 012737 000400 003214 MCV #HVRC,E.ER ;LOAD EXPECTED ERROR REG
4603 022764 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4604 022772 001401 BEQ 30$ ;YES, CONTINUE
4605 022774 104141 ERROR 141 ;CS1 INCORRECT
4606 022776 023737 003210 003150 30$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4607 023004 001401 BEQ 32$ ;YES, CONTINUE
4608 023006 104142 ERROR 142 ;CS2 INCORRECT
4609 023010 023737 003214 003154 32$: CMP E.ER,T.ER ;CHECK ERROR REG CORRECT
4610 023016 001401 BEQ 35$ ;YES - SKIP
4611 023020 104143 ERROR 143 ;ERROR REG INCORRECT
4612 023022 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4613 023030 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;GET CS1
4614 023036 016237 000010 003150 MOV RKCS2(R2),T.CS2 ; CS2
4615 023044 016237 000014 003154 MOV RKER(R2),T.ER ; ER
4616 023052 012737 000200 003200 MOV #200,E.CS1 ;SET EXPECTED CS1
4617 023060 023737 003140 003200 CMP T.CS1,E.CS1 ;CHECK IF CORRECT
4618 023066 001401 BEQ TST37 ;GO TO NEXT TEST
4619 023070 104153 ERROR 153 ;ERROR DID NOT CLEAR
  
```

```

*****
*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.
*****
  
```

```

4620
4621
4622
4623
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
4634 023072 000004 TST37: SCOPE
4635 023074 012737 000012 001200 MOV #10, $TIMES ;DO 10. ITERATIONS
4636 023102 013702 001270 001200 MOV $BASE,R2 ;LOAD RK611 BASE
4637 023106 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4638 023114 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4639 023122 012762 053606 000004 MOV #BUFF,RKBA(R2) ;LOAD DUMMY BUS ADDRESS
4640 023130 012762 177777 000002 MOV #-1,RKWC(R2) ;WORD COUNT=1
4641 023136 012762 000300 000020 MOV #300,RKDCYL(R2) ;LOAD CYLINDER 11DMS
4642 023144 012762 000000 000006 MOV #0,RKDA(R2) ;LOAD TRACK AND SECTOR
4643 023152 012762 000021 000000 MOV #RDATA,RKCS1(R2) ;ISSUE COMMAND
  
```

```

4644 023160 012700 000426      MOV      #69.*4+2,R0      ;ISSUE ENOUGH CLOCKS UNTIL
4645                                ;   READY FOR SECTOR PULSE.
4646 023164 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2)
4647 023172 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4648 023200 005300      DEC      R0
4649 023202 001370      BNE     1$
4650 023204 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4651 023212 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4652 023220 005037 003254      CLR     PR.BIT          ;GENERATE SYNCH
4653 023224 005037 003256      CLR     M1.BIT
4654 023230 012700 000377      MOV     #255.,R0
4655 023234 004737 035422      JSR     PC,RDBIT        5$:
4656 023240 005300      DEC     R0              ;CHECK IF SYNCH FINISHED
4657 023242 001374      BNE     5$
4658 023244 012737 000001 003254      MOV     #1,PR.BIT      ;SIMULATE SYNCH
4659 023252 004737 035422      JSR     PC,RDBIT
4660 023256 012703 052064      MOV     #HEAD10,R3     ;LOAD HEADER
4661 023262 012701 000003      MOV     #3,R1          ;LOAD WORDS PER HEADER
4662 023266 012304      MOV     (R3)+,R4       ;GET NEXT WORD
4663 023270 012700 000020      MOV     #16.,R0       ;LOAD BITS PER WORD
4664 023274 013737 003254 003256 12$:  MOV     PR.BIT,M1.BIT  ;STORE PREVIOUS BIT
4665 023302 006004      ROR     R4             ;GET NEXT BIT
4666 023304 103403      BCS     15$
4667 023306 005037 003254      CLR     PR.BIT
4668 023312 000403      BR      16$
4669
4670 023314 012737 000001 003254 15$:  MOV     #1,PR.BIT
4671 023322 004737 035422 16$:  JSR     PC,RDBIT      ;SIMULATE NEXT BIT
4672 023326 005300      DEC     R0             ;CHECK IF READY FOR NEXT WORD
4673 023330 001361      BNE     12$           ;NO, CONTINUE
4674 023332 005301      DEC     R1             ;CHECK IF FINISHED WITH HEADER
4675 023334 001354      BNE     10$           ;NO, CONTINUE
4676 023336 012700 000101      MOV     #65.,R0
4677 023342 013737 003254 003256 20$:  MOV     PR.BIT,M1.BIT  ;SIMULATE GAP
4678 023350 005037 003254      CLR     PR.BIT
4679 023354 004737 035422      JSR     PC,RDBIT
4680 023360 005300      DEC     R0             ;CHECK IF GAP FINISHED
4681 023362 001367      BNE     20$           ;NO, CONTINUE
4682 023364 012700 021200      MOV     #2*<256.+<16.*256.>+64.>,R0 ;LOAD COUNT UNTIL POSTAMBLE
4683 023370 012762 000440 000026 25$:  MOV     #DMD!MCLK,RKMR1(R2)
4684 023376 012762 000040 000026      MOV     #DMD,RKMR1(R2)
4685 023404 005300      DEC     R0
4686 023406 001370      BNE     25$
4687 023410 016237 000000 003140      MOV     RKCS1(R2),T.CS1 ;GET CS1
4688 023416 016237 000010 003150      MOV     RKCS2(R2),T.CS2 ;GET CS2
4689 023424 016237 000014 003154      MOV     RKER(R2),T.ER  ;GET ERROR REG
4690 023432 012737 100220 003200      MOV     #CERR!RDY!RDATA<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4691 023440 012737 000100 003210      MOV     #IR,E.CS2      ;LOAD EXPECTED CS2
4692 023446 012737 000400 003214      MOV     #HVRC,E.ER     ;LOAD EXPECTED ERROR REG
4693 023454 023737 003200 003140      CMP     E.CS1,T.CS1    ;CHECK CS1 CORRECT
4694 023462 001401      BEQ     30$           ;YES, CONTINUE
4695 023464 104141      ERROR  141            ;CS1 INCORRECT
4696 023466 023737 003210 003150 30$:  CMP     E.CS2,T.CS2    ;CHECK CS2 CORRECT
4697 023474 001401      BEQ     32$           ;YES, CONTINUE
4698 023476 104142      ERROR  142            ;CS2 INCORRECT
4699 023500 023737 003214 003154 32$:  CMP     E.ER,T.ER     ;CHECK ERROR REG CORRECT
  
```

```
4700 023506 001401 BEQ 35$ :YES - SKIP
4701 023510 104143 ERROR 143 :ERROR REG INCORRECT
4702 023512 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) :CLEAR CONTROLLER
4703 023520 016237 000000 003140 MOV RKCS1(R2),T.CS1 :GET CS1
4704 023526 016237 000010 003150 MOV RKCS2(R2),T.CS2 : CS2
4705 023534 016237 000014 003154 MOV RKER(R2),T.ER : ER
4706 023542 012737 000200 003200 MOV #200,E.CS1 :SET EXPECTED CS1
4707 023550 023737 003140 003200 CMP T.CS1,E.CS1 :CHECK IF CORRECT
4708 023556 001401 BEQ TST40 :GO TO NEXT TEST
4709 023560 104153 ERROR 153 :ERROR DID NOT CLEAR
```

```
4710
4711
4712 :*****
4713 :*TEST 40 WRITE CHECK AND HVRC
4714 :*
4715 :* CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4716 :* PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
4717 :* CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
4718 :* CYLINDER 300, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
4719 :* AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND
4720 :* A HEADER WITH A HEADER VRC ERROR. MAKE SURE
4721 :* HVRC AND CONTROLLER ERROR ARE SET. CLEAR CONTROLLER
4722 :* AND MAKE SURE CONTROLLER ERROR RESETS.
4723 :*****
```

```
4724 023562 000004 TST40: SCOPE
4725 023564 012737 000012 001200 MOV #10,$TIMES ;;DO 10. ITERATIONS
4726 023572 013702 001270 MOV $BASE,R2 :LOAD RK611 BASE
4727 023576 012762 100000 000000 MOV #CCLR,RKCS1(R2) :CLEAR RK611
4728 023604 012762 000040 000026 MOV #DMD,RKMR1(R2) :PUT RK611 IN DIAGNOSTIC MODE
4729 023612 012762 053606 000004 MOV #BUFF,RKBA(R2) :LOAD DUMMY BUS ADDRESS
4730 023620 012762 177777 000002 MOV #-1,RKWC(R2) :WORD COUNT=1
4731 023626 012762 000300 000020 MOV #300,RKDCYL(R2) :LOAD CYLINDER 11DMS
4732 023634 012762 000000 000006 MOV #0,RKDA(R2) :LOAD TRACK AND SECTOR
4733 023642 012762 000031 000000 MOV #WRTCHK,RKCS1(R2) :ISSUE COMMAND
4734 023650 012700 000426 MOV #69.*4+2,R0 :ISSUE ENOUGH CLOCKS UNTIL
4735 : READY FOR SECTOR PULSE.
4736 023654 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
4737 023662 012762 000040 000026 MOV #DMD,RKMR1(R2)
4738 023670 005300 DEC R0
4739 023672 001370 BNE 1$
4740 023674 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4741 023702 012762 000040 000026 MOV #DMD,RKMR1(R2)
4742 023710 005037 003254 CLR PR.BIT ;GENERATE SYNCH
4743 023714 005037 003256 CLR M1.BIT
4744 023720 012700 000377 MOV #255.,R0
4745 023724 004737 035422 5$: JSR PC,RDBIT
4746 023730 005300 DEC R0 ;CHECK IF SYNCH FINISHED
4747 023732 001374 BNE 5$
4748 023734 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH
4749 023742 004737 035422 JSR PC,RDBIT
4750 023746 012703 052064 MOV #HEAD10,R3 ;LOAD HEADER
4751 023752 012701 000003 MOV #3,R1 ;LOAD WORDS PER HEADER
4752 023756 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
4753 023760 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
4754 023764 013737 003254 12$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4755 023772 006004 ROR R4 ;GET NEXT BIT
```



```
4756 023774 103403          BCS 15$
4757 023776 005037 003254  CLR  PR.BIT
4758 024002 000403          BR   16$
4759
4760 024004 012737 000001 003254 15$: MOV #1,PR.BIT
4761 024012 004737 035422 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
4762 024016 005300          DEC  R0 ;CHECK IF READY FOR NEXT WORD
4763 024020 001361          BNE 12$ ;NO, CONTINUE
4764 024022 005301          DEC  R1 ;CHECK IF FINISHED WITH HEADER
4765 024024 001354          BNE 10$ ;NO, CONTINUE
4766 024026 012700 000101          MOV #65.,R0
4767 024032 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
4768 024040 005037 003254          CLR  PR.BIT
4769 024044 004737 035422          JSR  PC,RDBIT
4770 024050 005300          DEC  R0 ;CHECK IF GAP FINISHED
4771 024052 001367          BNE 20$ ;NO, CONTINUE
4772 024054 012700 021200          MOV #2*<256.+<16.*256.>+64.,R0 ;LOAD COUNT UNTIL POSTAMBLE
4773 024060 012762 000440 000026 25$: MOV #DMD!MCLK,RKMR1(R2)
4774 024066 012762 000040 000026          MOV #DMD,RKMR1(R2)
4775 024074 005300          DEC  R0
4776 024076 001370          BNE 25$
4777 024100 016237 000000 003140          MOV RKCS1(R2),T.CS1 ;GET CS1
4778 024106 016237 000010 003150          MOV RKCS2(R2),T.CS2 ;GET CS2
4779 024114 016237 000014 003154          MOV RKER(R2),T.ER ;GET ERROR REG
4780 024122 012737 100230 003200          MOV #CERR!RDY!WRTCHK<<C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4781 024130 012737 000100 003210          MOV #IR,E.CS2 ;LOAD EXPECTED CS2
4782 024136 012737 000400 003214          MOV #HVRC,E.ER ;LOAD EXPECTED ERROR REG
4783 024144 023737 003200 003140          CMP  E.CS1,T.CS1 ;CHECK CS1 CORRECT
4784 024152 001401          BEQ  30$ ;YES, CONTINUE
4785 024154 104141          ERROR 141 ;CS1 INCORRECT
4786 024156 023737 003210 003150 30$: CMP  E.CS2,T.CS2 ;CHECK CS2 CORRECT
4787 024164 001401          BEQ  32$ ;YES, CONTINUE
4788 024166 104142          ERROR 142 ;CS2 INCORRECT
4789 024170 023737 003214 003154 32$: CMP  E.ER,T.ER ;CHECK ERROR REG CORRECT
4790 024176 001401          BEQ  35$ ;YES - SKIP
4791 024200 104143          ERROR 143 ;ERROR REG INCORRECT
4792 024202 012762 100000 000000 35$: MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4793 024210 016237 000000 003140          MOV RKCS1(R2),T.CS1 ;GET CS1
4794 024216 016237 000010 003150          MOV RKCS2(R2),T.CS2 ; CS2
4795 024224 016237 000014 003154          MOV RKER(R2),T.ER ; ER
4796 024232 012737 000200 003200          MOV #200,E.CS1 ;SET EXPECTED CS1
4797 024240 023737 003140 003200          CMP  T.CS1,E.CS1 ;CHECK IF CORRECT
4798 024246 001401          BEQ  TST41 ;GO TO NEXT TEST
4799 024250 104153          ERROR 153 ;ERROR DID NOT CLEAR
```

4800  
4801  
4802  
4803  
4804  
4805  
4806  
4807  
4808  
4809  
4810  
4811

.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
```

```
4812      ;*      A SECTOR PULSE AND A GOOD HEADER.  CLOCK THROUGH ONLY FOUR
4813      ;*      DATA WORDS OF ZEROES.  MAKE SURE THE ECC PATTERN
4814      ;*      REGISTER REMAINS ZERO.
4815      ;*
4816      ;*
4817      ;*.....
4817 024252 000004      TST41: SCOPE
4818 024254 012737 000012 001200      MOV      #10.,$TIMES      ;;DO 10. ITERATIONS
4819 024262 013702 001270      MOV      $BASE,R2      ;:LOAD RK611 BASE
4820 024266 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;:CLEAR RK611
4821 024274 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
4822 024302 012762 052100 000004      MOV      #ECC1,RKBA(R2) ;:LOAD ADDRESS OF ECC DATA
4823 024310 012762 177770 000002      MOV      #-10,RKWC(R2) ;:WORD COUNT =10
4824 024316 012762 000023 000000      MOV      #WRDATA,RKCS1(R2) ;:ISSUE WRITE DATA
4825 024324 012700 000450      MOV      #8.*37.,RO      ;:ISSUE ENOUGH CLOCKS UNTIL
4826      ;:      READY FOR SECTOR PULSE
4827 024330 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2)
4828 024336 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4829 024344 005300      DEC      RO
4830 024346 001370      BNE     1$
4831 024350 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;:SIMULATE SECTOR PULSE
4832 024356 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4833 024364 005037 003254      CLR     PR.BIT      ;:GENERATE SYNCH
4834 024370 005037 003256      CLR     M1.BIT
4835 024374 012700 000377      MOV     #255.,RO
4836 024400 004737 035422      5$:      JSR     PC,RDBIT
4837 024404 005300      DEC     RO      ;:CHECK IF SYNCH FINISHED
4838 024406 001374      BNE     5$
4839 024410 012737 000001 003254      MOV     #1,PR.BIT      ;:SIMULATE SYNCH BIT
4840 024416 004737 035422      JSR     PC,RDBIT
4841 024422 012701 000003      MOV     #3,R1      ;:SIMULATE HEADER
4842 024426 012703 051754      MOV     #HEAD1,R3      ;:CYL 0, TRK 0, SECTOR 0
4843 024432 012304      12$:      MOV     (R3)+,R4      ;:GET NEXT HEADER WORD
4844 024434 012700 000020      MOV     #16.,RO      ;:LOAD BITS PER WORD
4845 024440 013737 003254 003256 15$:      MOV     PR.BIT,M1.BIT ;:STORE PREVIOUS BIT
4846 024446 006004      ROR     R4      ;:GET NEXT BIT
4847 024450 103403      BCS     17$
4848 024452 005037 003254      CLR     PR.BIT
4849 024456 000403      BR      18$
4850
4851 024460 012737 000001 003254 17$:      MOV     #1,PR.BIT
4852 024466 004737 035422 18$:      JSR     PC,RDBIT      ;:SIMULATE NEXT BIT
4853 024472 005300      DEC     RO      ;:CHECK IF READY FOR NEXT HEADER WORD
4854 024474 001361      BNE     15$      ;:NO, CONTINUE
4855 024476 005301      DEC     R1      ;:CHECK IF FINISHED WITH HEADER
4856 024500 001354      BNE     12$      ;:NO, CONTINUE
4857 024502 012700 000101      MOV     #65.,RO      ;:SIMULATE GAP +1 FOR SWITCH FROM READ TO WRITE
4858 024506 013737 003254 003256 20$:      MOV     PR.BIT,M1.BIT
4859 024514 005037 003254      CLR     PR.BIT
4860 024520 004737 035422      JSR     PC,RDBIT
4861 024524 005300      DEC     RO      ;:CHECK IF GAP FINISHED
4862 024526 001367      BNE     20$      ;:NO, CONTINUE
4863 024530 012700 000400      MOV     #256.,RO      ;:LOAD COUNT FOR SYNCH FIELD
4864 024534 012737 047260 001310      MOV     #EM327,EMW      ;:LOAD ERROR MESSAGE
4865 024542 005037 003252      CLR     P1.BIT      ;:INITIALIZE BITS
4866 024546 005037 003254      CLR     PR.BIT
4867 024552 005037 003256      CLR     M1.BIT
```

```
4868 024556 005037 003260 CLR M2.BIT
4869 024562 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
4870 024566 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;LOAD EXPECTED MR1
4871 024574 004737 034674 25$: JSR PC,WRTBIT ;WRITE BIT
4872 024600 104002 ERROR 2 ;DATA INCORRECT
4873 024602 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
4874 024606 005300 DEC R0 ;CHECK IF FINISHED
4875 024610 001371 BNE 25$ ;NO, CONTINUE
4876 024612 012737 000001 003252 MOV #1,P1.BIT ;SIMULATE SYNCH BIT
4877 024620 004737 034674 JSR PC,WRTBIT
4878 024624 104002 ERROR 2
4879 024626 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
4880 024632 012737 047332 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
4881 024640 005037 003234 CLR E.ECPT ;CLEAR EXPECTED ECC PATTERN
4882 024644 012703 052100 MOV #ECC1,R3 ;LOAD START OF DATA
4883 024650 012701 000004 MOV #4,R1 ;LOAD COUNT
4884 024654 012304 30$: MOV (R3)+,R4 ;GET NEXT WORD
4885 024656 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
4886 024662 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
4887 024670 013737 003254 003256 MOV PR.BIT,M1.BIT
4888 024676 013737 003252 003254 MOV P1.BIT,PR.BIT
4889 024704 006004 ROR R4 ;GET NEXT BIT
4890 024706 103403 BCS 34$
4891 024710 005037 003252 CLR P1.BIT
4892 024714 000403 BR 35$
4893
4894 024716 012737 000001 003252 34$: MOV #1,P1.BIT
4895 024724 004737 034674 35$: JSR PC,WRTBIT ;SIMULATE BIT
4896 024730 104002 ERROR 2
4897 024732 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;STORE ECC WORD
4898 024740 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
4899 024746 001401 BEQ 37$ ;YES, CONTINUE
4900 024750 104134 ERROR 134 ;ECC PATTERN NOT ZERO
4901 024752 005237 003262 37$: INC BITCNT ;INCREMENT BIT COUNT
4902 024756 005300 DEC R0 ;CHECK IF FINISHED WITH WORD
4903 024760 001340 BNE 32$ ;NO, CONTINUE
4904 024762 005301 DEC R1 ;CHECK IF FINISHED WITH TEST
4905 024764 001333 BNE 30$ ;NO, CONTINUE
```

```
4906
4907 *****
4908 *TEST 42 ECC GENERATION (PART 1)
4909 *
4910 * CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4911 * PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
4912 * DATA OF 12 WORDS TO AN RK06, IN 26 SECTOR
4913 * FORMAT, CYLINER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
4914 * SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR
4915 * PULSE AND A GOOD HEADER. CLOCK THROUGH ONLY THE
4916 * FOLLOWING SIX WORDS OF DATA:
4917 *
4918 * 005001
4919 * 040040
4920 * 020004
4921 * 000064
4922 * 000000
4923 * 000000
```

```

4924
4925
4926
4927
4928
4929 024766 000004
4930 024770 012737 000012 001200
4931 024776 013702 001270
4932 025002 012762 100000 000000
4933 025010 012762 000040 000026
4934 025016 012762 052120 000004
4935 025024 012762 177766 000002
4936 025032 012762 000023 000000
4937 025040 012700 000562
4938
4939 025044 012762 000440 000026 1$:
4940 025052 012762 000040 000026
4941 025060 005300
4942 025062 001370
4943 025064 012762 000140 000026
4944 025072 012762 000040 000026
4945 025100 005037 003254
4946 025104 005037 003256
4947 025110 012700 000377
4948 025114 004737 035422 5$:
4949 025120 005300
4950 025122 001374
4951 025124 012737 000001 003254
4952 025132 004737 035422
4953 025136 012701 000003
4954 025142 012703 051754
4955 025146 012304 12$:
4956 025150 012700 000020
4957 025154 013737 003254 003256 15$:
4958 025162 006004
4959 025164 103403
4960 025166 005037 003254
4961 025172 000403
4962
4963 025174 012737 000001 003254 17$:
4964 025202 004737 035422 18$:
4965 025206 005300
4966 025210 001361
4967 025212 005301
4968 025214 001354
4969 025216 012700 000101
4970 025222 013737 003254 003256 20$:
4971 025230 005037 003254
4972 025234 004737 035422
4973 025240 005300
4974 025242 001367
4975 025244 012700 000400
4976 025250 012737 047260 001310
4977 025256 005037 003252
4978 025262 005037 003254
4979 025266 005037 003256

```

\*\*\*\*\*  
TST42: 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 #ECC2,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  
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  
5\$: 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  
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 HEADER  
BNE 12\$ ;NO, CONTINUE  
MOV #65.,R0 ;SIMULATE GAP +1 FOR SWITCH FORM READ TO WRITE  
20\$: MOV PR.BIT,M1.BIT  
CLR PR.BIT  
JSR PC,RDBIT  
DEC R0 ;CHECK IF GAP FINISHED  
BNE 20\$ ;NO, CONTINUE  
MOV #256.,R0 ;LOAD COUNT FOR SYNCH FIELD  
MOV #EM327,EMW ;LOAD ERROR  
CLR P1.BIT ;INITIALIZE BITS  
CLR PR.BIT  
CLR M1.BIT

```
4980 025272 005037 003260 CLR M2.BIT
4981 025276 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
4982 025302 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
4983 025310 004737 034674 25$: JSR PC,WRTBIT ;WRITE BIT
4984 025314 104002 ERROR 2 ;DATA INCORRECT
4985 025316 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
4986 025322 005300 DEC R0 ;CHECK IF FINISHED
4987 025324 001371 BNE 25$ ;NO, CONTINUE
4988 025326 012737 000001 003252 MOV #1,P1.BIT ;SIMULATE SYNCH BIT
4989 025334 004737 034674 JSR PC,WRTBIT
4990 025340 104002 ERROR 2
4991 025342 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
4992 025346 012737 047332 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
4993 025354 005037 003266 CLR ECCHI ;INITIALIZE ECC
4994 025360 005037 003270 CLR ECCLO
4995 025364 012703 052120 MOV #ECC2,R3 ;LOAD START OF DATA
4996 025370 012701 000006 MOV #6,R1 ;LOAD COUNT
4997 025374 012304 30$: MOV (R3)+,R4 ;GET NEXT BIT
4998 025376 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
4999 025402 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5000 025410 013737 003254 003256 MOV PR.BIT,M1.BIT
5001 025416 013737 003252 003254 MOV P1.BIT,PR.BIT
5002 025424 006004 ROR R4 ;GET NEXT BIT
5003 025426 103403 BCS 34$
5004 025430 005037 003252 CLR P1.BIT
5005 025434 000403 BR 35$
5006
5007 025436 012737 000001 003252 34$: MOV #1,P1.BIT
5008 025444 004737 034674 35$: JSR PC,WRTBIT ;SIMULATE BIT
5009 025450 104002 ERROR 2
5010 025452 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;STORE ECC WORD
5011 025460 004737 034526 JSR PC,ECCGEN ;GENERATE EXPECTED ECC PAT
5012 025464 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
5013 025472 001401 BEQ 37$ ;YES, CONTINUE
5014 025474 104135 ERROR 135 ;ECC PATTERN INCORRECT
5015 025476 005237 003262 37$: INC BITCNT ;INCREMENT BIT COUNT
5016 025502 005300 DEC R0 ;CHECK IF FINISHED WITH WORD
5017 025504 001336 BNE 32$ ;NO, CONTINUE
5018 025506 005301 DEC R1 ;CHECK IF FINISHED WITH TEST
5019 025510 001331 BNE 30$ ;NO, CONTINUE
```

```
5020
5021
5022 :*****
5023 :*TEST 43 ECC GENERATION (PART 2)
5024 :*
5025 :* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5026 :* PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
5027 :* DATA OF 12 WORDS TO AN RK06, IN 26 SECTOR
5028 :* FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH
5029 :* SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE
5030 :* AND A GOOD HEADER. CLOCK THROUGH ONLY THE FOLLOWING
5031 :* SIX WORDS OF DATA:
5032 :*
5033 :* 177777
5034 :* 177777
5035 :* 177777
5036 :* 177777
```

```
5036          :*          177777
5037          :*          177777
5038          :*
5039          :*
5040          :*          CLOCK IN EACH BIT INDIVIDUALLY AND CHECK FOR PROPER ECC
5041          :*          GENERATION AS SEEN THROUGH THE ECC PATTERN REGISTER.
5042          :*
5043          :*****
5043 025512 000004 TST43: SCOPE
5044 025514 012737 000012 001200 MOV #10.,$TIMES ;;DO 10. ITERATIONS
5045 025522 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5046 025526 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5047 025534 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5048 025542 012762 052140 000004 MOV #ECC3,RKBA(R2) ;LOAD ECC DATA
5049 025550 012762 177766 000002 MOV #-12,RKWC(R2) ;WORD COUNT=12
5050 025556 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
5051 025564 012700 000562 MOV #10.*37.,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5052          : ; READY FOR SECTOR PULSE
5053 025570 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
5054 025576 012762 000040 000026 MOV #DMD,RKMR1(R2)
5055 025604 005300 DEC R0
5056 025606 001370 BNE 1$
5057 025610 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5058 025616 012762 000040 000026 MOV #DMD,RKMR1(R2)
5059 025624 005037 003254 CLR PR.BIT ;GENERATE SYNCH
5060 025630 005037 003256 CLR M1.BIT
5061 025634 012700 000377 MOV #255.,R0
5062 025640 004737 035422 5$: JSR PC,RDBIT
5063 025644 005300 DEC R0 ;CHECK IF SYNCH FINISHED
5064 025646 001374 BNE 5$
5065 025650 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH BIT
5066 025656 004737 035422 JSR PC,RDBIT
5067 025662 012701 000003 MOV #3,R1 ;SIMULATE HEADER
5068 025666 012703 051754 MOV #HEAD1,R3 ; CYL 0, TRK 0, SECTOR 0
5069 025672 012304 000020 12$: MOV (R3)+,R4 ;GET NEXT HEADER WORD
5070 025674 012700 000020 MOV #16.,R0 ;LOAD BITS PER WORD
5071 025700 013737 003254 003256 15$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
5072 025706 006004 ROR R4 ;GET NEXT BIT
5073 025710 103403 BCS 17$
5074 025712 005037 003254 CLR PR.BIT
5075 025716 000403 BR 18$
5076
5077 025720 012737 000001 003254 17$: MOV #1,PR.BIT
5078 025726 004737 035422 18$: JSR PC,RDBIT ;SIMULATE NEXT BIT
5079 025732 005300 DEC R0 ;CHECK IF READY FOR NEXT HEADER WORD
5080 025734 001361 BNE 15$ ;NO, CONTINUE
5081 025736 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
5082 025740 001354 BNE 12$ ;NO, CONTINUE
5083 025742 012700 000101 MOV #65.,R0 ;SIMULATE GAP +1 FOR SWITCH FORM READ TO WRITE
5084 025746 013737 003254 003256 20$: MOV PR.BIT,M1.BIT
5085 025754 005037 003254 CLR PR.BIT
5086 025760 004737 035422 JSR PC,RDBIT
5087 025764 005300 DEC R0 ;CHECK IF GAP FINISHED
5088 025766 001367 BNE 20$ ;NO, CONTINUE
5089 025770 012700 000400 MOV #256.,R0 ;LOAD COUNT FOR SYNCH FIELD
5090 025774 012737 047260 001310 MOV #EM327,EMW ;LOAD ERROR
5091 026002 005037 003252 CLR P1.BIT ;INITIALIZE BITS
```

```
5092 026006 005037 003254 CLR PR.BIT
5093 026012 005037 003256 CLR M1.BIT
5094 026016 005037 003260 CLR M2.BIT
5095 026022 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5096 026026 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5097 026034 004737 034674 25$: JSR PC,WRTBIT ;WRITE BIT
5098 026040 104002 ERROR 2 ;DATA INCORRECT
5099 026042 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
5100 026046 005300 DEC R0 ;CHECK IF FINISHED
5101 026050 001371 BNE 25$ ;NO, CONTINUE
5102 026052 012737 000001 003252 MOV #1,P1.BIT ;SIMULATE SYNCH BIT
5103 026060 004737 034674 JSR PC,WRTBIT
5104 026064 104002 ERROR 2
5105 026066 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
5106 026072 012737 047332 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
5107 026100 005037 003266 CLR ECCHI ;INITIALIZE ECC
5108 026104 005037 003270 CLR ECCLO
5109 026110 012703 052140 MOV #ECC3,R3 ;LOAD START OF DATA
5110 026114 012701 000006 MOV #6,R1 ;LOAD COUNT
5111 026120 012304 30$: MOV (R3)+,R4 ;GET NEXT BIT
5112 026122 012700 000020 MOV #16,R0 ;LOAD BITS PER WORD
5113 026126 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5114 026134 013737 003254 003256 MOV PR.BIT,M1.BIT
5115 026142 013737 003252 003254 MOV P1.BIT,PR.BIT
5116 026150 006004 ROR R4 ;GET NEXT BIT
5117 026152 103403 BCS 34$
5118 026154 005037 003252 CLR P1.BIT
5119 026160 000403 BR 35$
5120
5121 026162 012737 000001 003252 34$: MOV #1,P1.BIT
5122 026170 004737 034674 35$: JSR PC,WRTBIT ;SIMULATE BIT
5123 026174 104002 ERROR 2
5124 026176 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;STORE ECC WORD
5125 026204 004737 034526 JSR PC,ECCGEN ;GENERATE EXPECTED ECC PAT
5126 026210 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
5127 026216 001401 BEQ 37$ ;YES, CONTINUE
5128 026220 104135 ERROR 135 ;ECC PATTERN INCORRECT
5129 026222 005237 003262 37$: INC BITCNT ;INCREMENT BIT COUNT
5130 026226 005300 DEC R0 ;CHECK IF FINISHED WITH WORD
5131 026230 001336 BNE 32$ ;NO, CONTINUE
5132 026232 005301 DEC R1 ;CHECK IF FINISHED WITH TEST
5133 026234 001331 BNE 30$ ;NO, CONTINUE
```

```
5134
5135 *****
5136 :*TEST 44 ECC WRITING
5137 :*
5138 :* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5139 :* PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
5140 :* DATA OF 400 WORDS. TO AN RK06 IN 26 SECTOR FORMAT,
5141 :* CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
5142 :* AND DRIVE CLEAR MESSAGE. SIMULATE A SECTOR PULSE
5143 :* AND A GOOD HEADER. CLOCK THROUGH ALL 400 WORDS
5144 :* AND THE TWO ECC WORDS. MAKE SURE THE TWO ECC WORDS
5145 :* ARE CORRECT AND WRITTEN PROPERLY. CHECK BUS ADDRESS,
5146 :* WORD COUNT, CYLINDER, TRACK, AND SECTOR.
5147 :*
```

```
*****  
5148  
5149 026236 000004  
5150 026240 012737 000012 001200 1ST44: SCOPE  
5151  
5152 026246 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE  
5153 026252 012762 100000 000000 MOV #CLR,RKCS1(R2) ;CLEAR RK611  
5154 026260 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE  
5155 026266 012762 054606 000004 MOV #ECCBUF,RKBA(R2) ;LOAD ADDRESS  
5156 026274 012762 177400 000002 MOV #-400,RKWC(R2) ;WORD COUNT=400  
5157 026302 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA  
5158 026310 012700 004612 MOV #66.*37.,R0 ;ISSUE ENOUGH CLOCKS UNTIL  
5159 ; READY FOR SECTOR PULSE  
5160 026314 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)  
5161 026322 012762 000040 000026 MOV #DMD,RKMR1(R2)  
5162 026330 005300 DEC R0  
5163 026332 001370 BNE 1$  
5164 026334 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE  
5165 026342 012762 000040 000026 MOV #DMD,RKMR1(R2)  
5166 026350 005037 003254 CLR PR.BIT  
5167 026354 005037 003256 CLR M1.BIT  
5168 026360 012700 000377 MOV #255.,R0  
5169 026364 004737 035422 5$: JSR PC,RDBIT  
5170 026370 005300 DEC R0 ;CHECK IF SYNCH FINISHED  
5171 026372 001374 BNE 5$  
5172 026374 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH  
5173 026402 004737 035422 JSR PC,RDBIT  
5174 026406 012703 051754 MOV #HEAD1,R3 ;LOAD HEADER  
5175 026412 012701 000003 MOV #3,R1 ;LOAD WORDS PER HEADER  
5176 026416 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD  
5177 026420 012700 000020 MOV #16.,R0 ;LOAD BITS PER  
5178 026424 013737 003254 12$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT  
5179 026432 006004 ROR R4 ;GET NEXT BIT  
5180 026434 103403 BCS 15$  
5181 026436 005037 003254 CLR PR.BIT  
5182 026442 000403 BR 16$  
5183  
5184 026444 012737 000001 003254 15$: MOV #1,PR.BIT  
5185 026452 004737 035422 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT  
5186 026456 005300 DEC R0 ;CHECK IF READY FOR NEXT WORD  
5187 026460 001361 BNE 12$ ;NO, CONTINUE  
5188 026462 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER  
5189 026464 001354 BNE 10$ ;NO, CONTINUE  
5190 026466 012700 000101 MOV #65.,R0  
5191 026472 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP  
5192 026500 005037 003254 CLR PR.BIT  
5193 026504 004737 035422 JSR PC,RDBIT  
5194 026510 005300 DEC R0 ;CHECK IF GAP FINISHED  
5195 026512 001367 BNE 20$ ;NO, CONTINUE  
5196 026514 005037 003252 CLR P1.BIT ;INITIALIZE BITS FOR SYNCH  
5197 026520 005037 003254 CLR PR.BIT  
5198 026524 005037 003256 CLR M1.BIT  
5199 026530 005037 003260 CLR M2.BIT  
5200 026534 012737 047260 001310 MOV #EM327,EMW ;LOAD ERROR MESSAGE  
5201 026542 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT  
5202 026546 012700 000400 MOV #256.,R0 ;LOAD SYNCH COUNT AND WRITE SYNCH  
5203 026552 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
```



```

5204 026560 004737 034674      25$: JSR    PC,WRTBIT
5205 026564 104002             ERROR  2
5206 026566 005237 003262     INC    BITCNT      ;CLEAR BIT COUNT
5207 026572 005300             DEC    R0          ;CHECK IF SYNCH FINISHED
5208 026574 001371             BNE   25$         ;NO, CONTINUE
5209 026576 012737 000001 003252 MOV    #1,P1.BIT   ;WRITE SYNCH BIT
5210 026604 004737 034674     JSR    PC,WRTBIT
5211 026610 104002             ERROR  2
5212 026612 005037 003266     CLR    ECCHI      ;INITIALIZE ECC GENERATOR
5213 026616 005037 003270     CLR    ECCLO
5214 026622 012737 047332 001310 MOV    #EM328,EMW  ;LOAD ERROR MESSAGE
5215 026630 005037 003262     CLR    BITCNT     ;INITIALIZE BIT COUNT
5216 026634 012701 000400     MOV    #256.,R1   ;LOAD WORDS PER SECTOR
5217 026640 012703 054606     MOV    #ECCBUF,R3 ;LOAD ADDRESS OF BUFFER
5218 026644 012304             MOV    (R3)+,R4   ;GET NEXT WORD
5219 026646 012700 000020             MOV    #16.,R0   ;LOAD BITS PER WORD
5220 026652 013737 003256 003260 32$: MOV    M1.BIT,M2.BIT ;SHIFT BITS
5221 026660 013737 003254 003256 MOV    PR.BIT,M1.BIT
5222 026666 013737 003252 003254 MOV    P1.BIT,PR.BIT
5223 026674 006004             ROR    R4         ;DETERMINE NEXT BIT
5224 026676 103403             BCS   34$
5225 026700 005037 003252     CLR    P1.BIT
5226 026704 000403             BR    35$
5227
5228 026706 012737 000001 003252 34$: MOV    #1,P1.BIT
5229 026714 004737 034674     35$: JSR    PC,WRTBIT   ;WRITE NEXT BIT
5230 026720 104002             ERROR  2
5231 026722 004737 034526     JSR    PC,ECCGEN  ;GENERATE ECC
5232 026726 016237 000032 003174 MOV    RKECPT(R2),T.ECPT ;GET PATTERN
5233 026734 023737 003234 003174 CMP    E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
5234 026742 001401             BEQ   37$         ;YES, CONTINUE
5235 026744 104135             ERROR  135        ;ECC PATTERN INCORRECT
5236 026746 005237 003262     37$: INC    BITCNT     ;INCREMENT BIT COUNT
5237 026752 022700 000002     CMP    #2,R0     ;IF THE NEXT BIT IS THE LAST BIT OF
5238 026756 001006             BNE   28$        ;THE LAST DATA WORD, ECCW MUST BE RESET
5239 026760 022701 000001     CMP    #1,R1     ;IN THE EXPECTED MR1 TO INDICATE ECC
5240 026764 001003             BNE   28$        ;BEING WRITTEN
5241 026766 042737 020000 003224 BIC    #ECCW,E.MR1
5242 026774 005300             28$: DEC    R0         ;CHECK IF THROUGH WITH WORD
5243 026776 001325             BNE   32$        ;NO, CONTINUE
5244 027000 005301             DEC    R1         ;CHECK IF AT END OF SECTOR
5245 027002 001320             BNE   30$        ;NO, CONTINUE
5246 027004 012737 047677 001310 MOV    #EM333,EMW  ;LOAD ERROR MESSAGE
5247 027012 005037 003262     CLR    BITCNT     ;INITIALIZE BIT COUNT
5248 027016 012701 000002     MOV    #2,R1     ;LOAD NUMBER OF ECO WORDS
5249 027022 012703 003266     MOV    #ECCHI,R3  ;LOAD ADDRESS OF ECC
5250 027026 012304             40$: MOV    (R3)+,R4   ;GET NEXT ECC WORD
5251 027030 012700 000020             MOV    #16.,R0   ;LOAD BITS PER WORD
5252 027034 013737 003256 003260 42$: MOV    M1.BIT,M2.BIT ;SHIFT BITS
5253 027042 013737 003254 003256 MOV    PR.BIT,M1.BIT
5254 027050 013737 003252 003254 MOV    P1.BIT,PR.BIT
5255 027056 006004             ROR    R4         ;DETERMINE NEXT BIT
5256 027060 103403             BCS   44$
5257 027062 005037 003252     CLR    P1.BIT
5258 027066 000403             BR    45$
5259

```

```

5260 027070 012737 000001 003252 44$: MOV #1,P1.BIT
5261 027076 004737 034674 45$: JSR PC,WRTBIT ;WRITE NEXT BIT
5262 027102 104002 ERROR 2
5263 027104 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
5264 027110 022700 000002 CMP #2,RO ;IF THE LAST BIT OF THE LAST ECC WORD
5265 027114 001006 BNE 46$ ;IS BEING WRITTEN, ECCW MUST BE SET
5266 027116 022701 000001 CMP #1,R1 ;IN EXPECTED MR1 TO INDICATE ECC
5267 027122 001003 BNE 46$ ;WRITING IS DONE
5268 027124 052737 020000 003224 BIS #ECCW,E.MR1
5269 027132 005300 46$: DEC RO ;CHECK IF THROUGH WITH WORD
5270 027134 001337 BNE 42$ ;NO, CONTINUE
5271 027136 005301 DEC R1 ;CHECK IF FINISH WITH ECC
5272 027140 001332 BNE 40$ ;NO, CONTINUE
5273 027142 012737 047735 001310 MOV #EM334,EMW ;LOAD ERROR MESSAGE
5274 027150 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
5275 027154 012700 000017 MOV #15,RO ;LOAD POSTAMBLE BIT COUNT
5276 027160 013737 003256 003260 47$: MOV M1.BIT,M2.BIT ;SHIFT BIT
5277 027166 013737 003254 003256 MOV PR.BIT,M1.BIT
5278 027174 013737 003252 003254 MOV P1.BIT,PR.BIT
5279 027202 005037 003252 CLR P1.BIT
5280 027206 004737 034674 JSR PC,WRTBIT ;WRITE NEXT BIT
5281 027212 104002 ERROR 2
5282 027214 005237 003262 INC BITCNT ;INCREMENT BIT COUNT
5283 027220 005300 DEC RO ;CHECK IF THROUGH WITH POSTAMBLE
5284 027222 001356 BNE 47$ ;NO, CONTINUE
5285 027224 012700 000014 MOV #3*4,RO ;ISSUE CLOCKS TO COMPLETE COMMAND
5286 027230 012762 000440 000026 50$: MOV #DMD!MCLK,RKMR1(R2) ;ISSUE CLOCK PULSES
5287 027236 012762 000040 000026 MOV #DMD,RKMR1(R2)
5288 027244 005300 DEC RO
5289 027246 001370 BNE 50$
5290 027250 016237 000000 003140 MOV RKCS1(R2),T.CS1 ;SAVE CS1
5291 027256 016237 000010 003150 MOV RKCS2(R2),T.CS2 ;SAVE CS2
5292 027264 016237 000014 003154 MOV RKER(R2),T.ER ;SAVE ERROR REG
5293 027272 012737 000222 003200 MOV #RDY!WRDATA&<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5294 027300 012737 000100 003210 MOV #1R,E.CS2 ;LOAD EXPECTED CS2
5295 027306 005037 003214 CLR E.ER ;LOAD EXPECTED ERROR REG.
5296 027312 016237 000020 003160 MOV RKDCYL(R2),T.DCYL ;SAVE CYLINDER ADDR REG
5297 027320 016237 000006 003146 MOV RKDA(R2),T.DA ;SAVE DISK AND REG
5298 027326 016237 000004 003144 MOV RKBA(R2),T.BA ;SAVE BUS ADDR REG
5299 027334 016237 000002 003142 MOV RKWC(R2),T.WC ;SAVE WORD COUNT
5300 027342 005037 003220 CLR E.DCYL ;LOAD EXPECTED CYLINDER AND REG.
5301 027346 012737 000001 003206 MOV #1,E.DA ;LOAD EXPECTED DISK
5302 027354 012737 055606 003204 MOV #ECCBUF+<400*2>,E.BA ;LOAD EXPECTED
5303 027362 005037 003202 CLR E.WC ;LOAD EXPECTED ECC WORD
5304 027366 023737 003200 003140 CMP E.CS1,T.CS1 ;CHECK CS1
5305 027374 001401 BEQ 55$
5306 027376 104144 ERROR 144
5307 027400 023737 003210 003150 55$: CMP E.CS2,T.CS2 ;CHECK CS2
5308 027406 001401 BEQ 56$
5309 027410 104145 ERROR 145
5310 027412 023737 003214 003154 56$: CMP E.ER,T.ER ;CHECK ERROR REG
5311 027420 001401 BEQ 57$
5312 027422 104146 ERROR 146
5313 027424 023737 003204 003144 57$: CMP E.BA,T.BA ;CHECK BUS ADD
5314 027432 001401 BEQ 58$
5315 027434 104147 ERROR 147

```

```

5316 027436 023737 003202 003142 58$: CMP E.WC,T.WC ;CHECK WORD COUNT
5317 027444 001401 BEQ 59$
5318 027446 104150 ERROR 150
5319 027450 023737 003220 003160 59$: CMP E.DCYL,T.DCYL ;CHECK CYLINDER ADD
5320 027456 001401 BEQ 60$
5321 027460 104151 ERROR 151
5322 027462 023737 003206 003146 60$: CMP E.DA,T.DA ;CHECK DISK ADDR.
5323 027470 001401 BEQ TST45 ;:YES, GO ON TO NEXT TEST
5324 027472 104152 ERROR 152
  
```

.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.
:*****
  
```

```

5341
5342 027474 000004 TST45: SCOPE
5343 027476 012737 000012 001200 MOV #10.,$TIMES ;;DO 10. ITERATIONS
5344
5345 027504 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5346 027510 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5347 027516 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5348 027524 012762 054606 000004 MOV #ECCBUF,RKBA(R2) ;LOAD ADDRESS
5349 027532 012762 177675 000002 MOV #-103,RKWC(R2) ;WORD COUNT=400
5350 027540 012762 000023 000000 MOV #WRDATA,RKCS1(R2) ;ISSUE WRITE DATA
5351 027546 012700 004612 MOV #66.*37.,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5352 ; READY FOR SECTOR PULSE
5353 027552 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
5354 027560 012762 000040 000026 MOV #DMD,RKMR1(R2)
5355 027566 005300 DEC R0
5356 027570 001370 BNE 1$
5357 027572 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5358 027600 012762 000040 000026 MOV #DMD,RKMR1(R2)
5359 027606 005037 003254 CLR PR.BIT
5360 027612 005037 003256 CLR M1.BIT
5361 027616 012700 000377 MOV #255.,R0
5362 027622 004737 035422 5$: JSR PC,RDBIT
5363 027626 005300 DEC R0 ;CHECK IF SYNCH FINISHED
5364 027630 001374 BNE 5$
5365 027632 012737 000001 003254 MOV #1,PR.BIT ;SIMULATE SYNCH
5366 027640 004737 035422 JSR PC,RDBIT
5367 027644 012703 051754 MOV #HEAD1,R3 ;LOAD HEADER
5368 027650 012701 000003 MOV #3,R1 ;LOAD WORDS PER HEADER
5369 027654 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
5370 027656 012700 000020 MOV #16.,R0 ;LOAD BITS PER
5371 027662 013737 003254 12$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
  
```

```

5372 027670 006004 ROR R4 ;GET NEXT BIT
5373 027672 103403 BCS 15$
5374 027674 005037 003254 CLR PR.BIT
5375 027700 000403 BR 16$
5376
5377 027702 012737 000001 003254 15$: MOV #1,PR.BIT
5378 027710 004737 035422 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
5379 027714 005300 DEC R0 ;CHECK IF READY FOR NEXT WORD
5380 027716 001361 BNE 12$ ;NO, CONTINUE
5381 027720 005301 DEC R1 ;CHECK IF FINISHED WITH HEADER
5382 027722 001354 BNE 10$ ;NO, CONTINUE
5383 027724 012700 000101 MOV #65.,R0
5384 027730 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
5385 027736 005037 003254 CLR PR.BIT
5386 027742 004737 035422 JSR PC,RDBIT
5387 027746 005300 DEC R0 ;CHECK IF GAP FINISHED
5388 027750 001367 BNE 20$ ;NO, CONTINUE
5389 027752 005037 003252 CLR P1.BIT ;INITIALIZE BITS FOR SYNCH
5390 027756 005037 003254 CLR PR.BIT
5391 027762 005037 003256 CLR M1.BIT
5392 027766 005037 003260 CLR M2.BIT
5393 027772 012737 047260 001310 MOV #EM327,EMW ;LOAD ERROR MESSAGE
5394 030000 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5395 030004 012700 000400 MOV #256.,R0 ;LOAD SYNCH COUNT AND WRITE SYNCH
5396 030010 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5397 030016 004737 034674 25$: JSR PC,WRTBIT
5398 030022 104002 ERROR 2
5399 030024 005237 003262 INC BITCNT ;CLEAR BIT COUNT
5400 030030 005300 DEC R0 ;CHECK IF SYNCH FINISHED
5401 030032 001371 BNE 25$ ;NO, CONTINUE
5402 030034 012737 000001 003252 MOV #1,P1.BIT ;WRITE SYNCH BIT
5403 030042 004737 034674 JSR PC,WRTBIT
5404 030046 104002 ERROR 2
5405 030050 005037 003266 CLR ECCHI ;INITIALIZE ECC GENERATOR
5406 030054 005037 003270 CLR ECCLO
5407 030060 012737 047332 001310 MOV #EM328,EMW ;LOAD ERROR MESSAGE
5408 030066 005037 003262 CLR BITCNT ;INITIALIZE BIT COUNT
5409 030072 012701 000400 MOV #256.,R1 ;LOAD WORDS PER SECTOR
5410 030076 012703 054606 MOV #ECCBUF,R3 ;LOAD ADDRESS OF BUFFER
5411 030102 012304 30$: MOV (R3)+,R4 ;GET NEXT WORD
5412 030104 012700 000020 31$: MOV #16.,R0 ;LOAD BITS PER WORD
5413 030110 013737 003256 003260 32$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5414 030116 013737 003254 003256 MOV PR.BIT,M1.BIT
5415 030124 013737 003252 003254 MOV P1.BIT,PR.BIT
5416 030132 006004 ROR R4 ;DETERMINE NEXT BIT
5417 030134 103403 BCS 34$
5418 030136 005037 003252 CLR P1.BIT
5419 030142 000403 BR 35$
5420
5421 030144 012737 000001 003252 34$: MOV #1,P1.BIT
5422 030152 004737 034674 35$: JSR PC,WRTBIT ;WRITE NEXT BIT
5423 030156 104002 ERROR 2
5424 030160 004737 034526 JSR PC,ECCGEN ;GENERATE ECC
5425 030164 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;GET PATTERN
5426 030172 023737 003234 003174 CMP E.ECPT,T.ECPT ;CHECK ECC PATTERN CORRECT
5427 030200 001401 BEQ 37$ ;YES, CONTINUE

```

5428	030202	104135				ERROR	135	;ECC PATTERN INCORRECT
5429	030204	005237	003262	37\$:		INC	BITCNT	;INCREMENT BIT COUNT
5430	030210	022700	000002			CMP	#2,RO	;IF THE NEXT BIT IS THE LAST BIT OF
5431	030214	001006				BNE	28\$	;THE LAST DATA WORD, ECCW MUST BE RESET
5432	030216	022701	000001			CMP	#1,R1	;IN THE EXPECTED MR1 TO INDICATE ECC
5433	030222	001003				BNE	28\$	;BEING WRITTEN
5434	030224	042737	020000	003224		BIC	#ECCW,E.MR1	
5435	030232	005300			28\$:	DEC	RO	;CHECK IF THROUGH WITH WORD
5436	030234	001325				BNE	32\$	;NO, CONTINUE
5437	030236	005301				DEC	R1	;CHECK IF AT END OF SECTOR
5438	030240	001405				BEQ	39\$	;YES -SKIP
5439	030242	022703	055014			CMP	#ECCBUF+<103*2>,R3	;CHECK IF 103 WORDS TRANSFERED
5440	030246	003315				BGT	30\$	;NO - GO TO GET NEXT WORD
5441	030250	005004				CLR	R4	;ELSE CLEAR R4 FOR ZEROS
5442	030252	000714				BR	31\$	;GO SIMULATE REST OF SECTOR
5443	030254	012737	047677	001310	39\$:	MOV	#EM333,EMW	;LOAD ERROR MESSAGE
5444	030262	005037	003262			CLR	BITCNT	;INITIALIZE BIT COUNT
5445	030266	012701	000002			MOV	#2,R1	;LOAD NUMBER OF ECO WORDS
5446	030272	012703	003266			MOV	#ECCHI,R3	;LOAD ADDRESS OF ECC
5447	030276	012304			40\$:	MOV	(R3)+,R4	;GET NEXT ECC WORD
5448	030300	012700	000020			MOV	#16.,RO	;LOAD BITS PER WORD
5449	030304	013737	003256	003260	42\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
5450	030312	013737	003254	003256		MOV	PR.BIT,M1.BIT	
5451	030320	013737	003252	003254		MOV	P1.BIT,PR.BIT	
5452	030326	006004				ROR	R4	;DETERMINE NEXT BIT
5453	030330	103403				BCS	44\$	
5454	030332	005037	003252			CLR	P1.BIT	
5455	030336	000403				BR	45\$	
5456								
5457	030340	012737	000001	003252	44\$:	MOV	#1,P1.BIT	
5458	030346	004737	034674		45\$:	JSR	PC,WRTBIT	;WRITE NEXT BIT
5459	030352	104002				ERROR	2	
5460	030354	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
5461	030360	022700	000002			CMP	#2,RO	;IF THE LAST BIT OF THE LAST ECC WORD
5462	030364	001006				BNE	46\$	;IS BEING WRITTEN, ECCW MUST BE SET
5463	030366	022701	000001			CMP	#1,R1	;IN EXPECTED MR1 TO INDICATE ECC
5464	030372	001003				BNE	46\$	;WRITING IS DONE
5465	030374	052737	020000	003224		BIS	#ECCW,E.MR1	
5466	030402	005300			46\$:	DEC	RO	;CHECK IF THROUGH WITH WORD
5467	030404	001337				BNE	42\$	;NO, CONTINUE
5468	030406	005301				DEC	R1	;CHECK IF FINISH WITH ECC
5469	030410	001332				BNE	40\$	;NO, CONTINUE
5470	030412	012737	047735	001310		MOV	#EM334,EMW	;LOAD ERROR MESSAGE
5471	030420	005037	003262			CLR	BITCNT	;CLEAR BIT COUNT
5472	030424	012700	000017			MOV	#15.,RO	;LOAD POSTAMBLE BIT COUNT
5473	030430	013737	003256	003260	47\$:	MOV	M1.BIT,M2.BIT	;SHIFT BIT
5474	030436	013737	003254	003256		MOV	PR.BIT,M1.BIT	
5475	030444	013737	003252	003254		MOV	P1.BIT,PR.BIT	
5476	030452	005037	003252			CLR	P1.BIT	
5477	030456	004737	034674			JSR	PC,WRTBIT	;WRITE NEXT BIT
5478	030462	104002				ERROR	2	
5479	030464	005237	003262			INC	BITCNT	;INCREMENT BIT COUNT
5480	030470	005300				DEC	RO	;CHECK IF THROUGH WITH POSTAMBLE
5481	030472	001356				BNE	47\$	;NO, CONTINUE
5482	030474	012700	000014			MOV	#3*4,RO	;ISSUE CLOCKS TO COMPLETE COMMAND
5483	030500	012762	000440	000026	50\$:	MOV	#DMD!MCLK,RKMR1(R2)	;ISSUE CLOCK PULSES

```

5484 030506 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5485 030514 005300                      DEC      R0
5486 030516 001370                      BNE     50$
5487 030520 016237 000000 003140      MOV      RKCS1(R2),T.CS1 ;SAVE CS1
5488 030526 016237 000010 003150      MOV      RKCS2(R2),T.CS2 ;SAVE CS2
5489 030534 016237 000014 003154      MOV      RKER(R2),T.ER   ;SAVE ERROR REG
5490 030542 012737 000222 003200      MOV      #RDY!WRDATA<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5491 030550 012737 000100 003210      MOV      #IR,E.CS2      ;LOAD EXPECTED CS2
5492 030556 005037 003214                      CLR     E.ER            ;LOAD EXPECTED ERROR REG.
5493 030562 016237 000020 003160      MOV      RKDCYL(R2),T.DCYL ;SAVE CYLINDER ADDR REG
5494 030570 016237 000006 003146      MOV      RKDA(R2),T.DA   ;SAVE DISK AND REG
5495 030576 016237 000004 003144      MOV      RKBA(R2),T.BA   ;SAVE BUS ADD REG
5496 030604 016237 000002 003142      MOV      RKWC(R2),T.WC   ;SAVE WORD COUNT
5497 030612 005037 003220                      CLR     E.DCYL         ;LOAD EXPECTED CYLINDER AND REG.
5498 030616 012737 000001 003206      MOV      #1,E.DA        ;LOAD EXPECTED DISK
5499 030624 012737 055014 003204      MOV      #ECCBUF+<103*2>,E.BA ;LOAD EXPECTED
5500 030632 005037 003202                      CLR     E.WC           ;LOAD EXPECTED ECC WORD
5501 030636 023737 003200 003140      CMP      E.CS1,T.CS1    ;CHECK CS1
5502 030644 001401                      BEQ     55$
5503 030646 104154                      ERROR   154
5504 030650 023737 003210 003150 55$:  CMP      E.CS2,T.CS2    ;CHECK CS2
5505 030656 001401                      BEQ     56$
5506 030660 104155                      ERROR   155
5507 030662 023737 003214 003154 56$:  CMP      E.ER,T.ER      ;CHECK ERROR REG
5508 030670 001401                      BEQ     57$
5509 030672 104156                      ERROR   156
5510 030674 023737 003204 003144 57$:  CMP      E.BA,T.BA      ;CHECK BUS ADD
5511 030702 001401                      BEQ     58$
5512 030704 104157                      ERROR   157
5513 030706 023737 003202 003142 58$:  CMP      E.WC,T.WC      ;CHECK WORD COUNT
5514 030714 001401                      BEQ     59$
5515 030716 104160                      ERROR   160
5516 030720 023737 003220 003160 59$:  CMP      E.DCYL,T.DCYL  ;CHECK CYLINDER ADD
5517 030726 001401                      BEQ     60$
5518 030730 104161                      ERROR   161
5519 030732 023737 003206 003146 60$:  CMP      E.DA,T.DA      ;CHECK DISK ADDR.
5520 030740 001401                      BEQ     TST46           ;;YES, GO ON TO NEXT TEST
5521 030742 104162                      ERROR   162

```

.SBTTL \*\*18 BIT FORMAT WRITES

:NOTE: SINCE 18-BIT FUNCTIONALITY OF THE RK611 IS NOT  
 :UTILIZED ON THE PDP-11, THE FOLLOWING TESTS, INTENDED  
 :FOR MANUFACTURING USE, ARE NOT EXECUTED UNLESS LOCA-  
 :TION "SEC24" IS PATCHED TO A NON-ZERO VALUE.

\*\*\*\*\*  
 :\*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
:* 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.
:*

```

5522  
5523  
5524  
5525  
5526  
5527  
5528  
5529  
5530  
5531  
5532  
5533  
5534  
5535  
5536  
5537  
5538  
5539

```

5540 .....
5541 030744 000004          TST46: SCOPE
5542 030746 012737 000012 001200 MOV #10.,$TIMES      ;;DO 10. ITERATIONS
5543 030754 005737 003314          TST SEC24          ;;SKIP 18 BIT TESTS?
5544 030760 001002          BNE 1$             ;;BRANCH IF NOT
5545 030762 000137 032760          JMP SEOP           ;;ELSE GO TO END OF PASS
5546 030766          1$:
5547
5548 030766 013702 001270          MOV $BASE,R2      ;LOAD RK611 BASE
5549 030772 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5550 031000 012700 002000          MOV #2000,R0      ;SET A STALL COUNT
5551 031004 005300          5$: DEC R0          ;LOOP UNTIL COUNT 0
5552 031006 001376          BNE 5$
5553
5554 031010 012762 000040 000026 MOV #DMD,RKMR1(R2) ;SET DIAGNOSTIC MODE
5555 031016 012762 053572 000004 MOV #MOD2BF,RKBA(R2) ;LOAD BUFFER ADDRESS
5556 031024 012762 177400 000002 MOV #-400,RKWC(R2) ; --- WORD COUNT
5557 031032 012762 010023 000000 MOV #WRDATA!CFMT,RKCS1(R2) ; --- START COMMAND
5558 031040 012700 005136          MOV #66.*40.+<4.*3.+2>,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5559          ; READY FOR SECTOR PULSE
5560 031044 012762 000440 000026 7$: MOV #DMD!MCLK,RKMR1(R2)
5561 031052 012762 000040 000026 MOV #DMD,RKMR1(R2)
5562 031060 005300          DEC R0
5563 031062 001370          BNE 7$
5564
5565 031064 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5566 031072 012762 000040 000026 MOV #DMD,RKMR1(R2)
5567
5568 031100 005037 003254          CLR PR.BIT        ;GENERATE SYNC
5569 031104 005037 003256          CLR M1.BIT
5570 031110 012700 000377          MOV #255.,R0
5571
5572 031114 004737 035422          9$: JSR PC,RDBIT    ;SIMULATE SYNC 0 BITS
5573 031120 005300          DEC R0
5574 031122 001374          BNE 9$
5575
5576 031124 012737 000001 003254 MOV #1,PR.BIT      ;SIMULATE SYNC 1 BIT
5577 031132 004737 035422          JSR PC,RDBIT
5578
5579 031136 012703 052072          MOV #HEAD11,R3    ;SIMULATE HEADER
5580 031142 012701 000003          MOV #3,R1         ; CYL 0 TRK 0, SEC 0
5581 031146 012304          11$: MOV (R3)+,R4      ;GET HEADER WORD
5582 031150 012700 000020          MOV #16.,R0       ;LOAD BITS PER WORD
5583 031154 013737 003254 003256 13$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
5584 031162 006004          ROR R4           ;GET NEXT BIT
5585 031164 103403          BCS 15$
5586 031166 005037 003254          CLR PR.BIT
5587 031172 000403          BR 16$
5588
5589 031174 012737 000001 003254 15$: MOV #1,PR.BIT
5590 031202 004737 035422 16$: JSR PC,RDBIT    ;SIMULATE NEXT BIT
5591 031206 005300          DEC R0           ;READY FOR NEXT HEADER WORD?
5592 031210 001361          BNE 13$         ;NO - GET NEXT BIT THIS WORD
5593 031212 005301          DEC R1          ;HEADER DONE?
5594 031214 001354          BNE 11$         ;NO - GET NEXT HEADER WORD
5595 031216 012700 000101          MOV #65.,R0      ;SET COUNT FOR GAP.

```

```

5596 031222 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
5597 031230 005037 003254 CLR PR.BIT
5598 031234 004737 035422 JSR PC,RDBIT
5599 031240 005300 DEC R0
5600 031242 001367 BNE 20$
5601 031244 012700 000400 MOV #256,R0 ;SET COUNT FOR WRITE DATA SYNC
5602 031250 012737 047260 001310 MOV #EM327,EMW ;LOAD ERROR MESSAGE
5603 031256 005037 003252 CLR P1.BIT ;CLEAR BITS
5604 031262 005037 003254 CLR PR.BIT
5605 031266 005037 003256 CLR M1.BIT
5606 031272 005037 003260 CLR M2.BIT
5607 031276 005037 003262 CLR BITCNT ;CLEAR BIT COUNTER
5608 031302 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5609 031310 004737 034674 22$: JSR PC,WRTBIT ;SIMULATE SYNC 0
5610 031314 104002 ERROR 2
5611 031316 005237 003262 INC BITCNT ;BUMP BIT COUNT
5612 031322 005300 DEC R0 ;LOOP UNTIL SYNC 0 WRITTEN
5613 031324 001371 BNE 22$
5614 031326 012737 000001 003252 MOV #1,P1.BIT ;SIMULATE SYNC 1
5615 031334 004737 034674 JSR PC,WRTBIT
5616 031340 104002 ERROR 2
5617 031342 005037 003266 CLR ECCHI ;INITIALIZE ECC WORDS.
5618 031346 005037 003270 CLR ECCLO
5619 031352 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
5620 031356 012703 053572 MOV #MOD2BF,R3 ;SET DATA POINTER
5621 031362 012701 000006 MOV #6,R1 ;SET DATA COUNT
5622 031366 012304 24$: MOV (R3)+,R4 ;GET DATA WORD
5623 031370 012737 050203 001310 MOV #EM338,EMW ;LOAD MESSAGE (18 BIT DATA WRITE)
5624 031376 012700 000022 MOV #18,R0 ;LOAD BITS PER WORD
5625 031402 013737 003256 003260 25$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5626 031410 013737 003254 003256 MOV PR.BIT,M1.BIT
5627 031416 013737 003252 003254 MOV P1.BIT,PR.BIT
5628 031424 000241 CLC ;CLEAR CARRY & GET NEXT BIT
5629 031426 006004 ROR R4
5630 031430 103403 BCS 27$
5631 031432 005037 003252 CLR P1.BIT
5632 031436 000403 BR 28$
5633
5634 031440 012737 000001 003252 27$: MOV #1,P1.BIT
5635 031446 004737 034674 28$: JSR PC,WRTBIT ;SIMULATE NEXT BIT
5636 031452 104002 ERROR 2
5637 031454 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;GET ECC PATTERN
5638 031462 004737 034526 JSR PC,ECCGEN ;COMPUTE EXPECTED ECC
5639 031466 023737 003174 003234 CMP T.ECPT,E.ECPT ;CHECK IF CORRECT
5640 031474 001401 BEQ 29$ ;YES - SKIP
5641 031476 104164 ERROR 164
5642 031500 005237 003262 29$: INC BITCNT
5643 031504 005300 DEC R0
5644 031506 020027 000002 CMP R0,#2 ;1ST 16 BITS JUST DONE?
5645 031512 001403 BEQ 31$ ;YES - SKIP
5646 031514 005700 TST R0 ;ELSE TEST IF WORD DONE
5647 031516 001331 BNE 25$ ;NO - DO NEXT BIT
5648 031520 000406 BR 32$ ;ELSE DO NEXT WORD
5649 031522 012737 050271 001310 31$: MOV #EM339,EMW ;LOAD MESSAGE (BIT 16.17)
5650 031530 012704 000000 MOV #0,R4 ;SET UPPER BITS OF WORD
5651 031534 000722 BR 25$ ;GO DO BITS

```



```

5652
5653 031536 005301      32$:  DEC      R1      ;ALL WORDS DONE?
5654 031540 001312      BNE      24$      ;NO - GET NEXT WORD
5655
5656
5657
5658
5659
5660
5661
5662
5663
5664
5665
5666
5667
5668
5669 031542 000004      TST47: SCOPE
5670 031544 012737 000012 001200      MOV      #10.,$TIMES      ;;DO 10. ITERATIONS
5671
5672
5673 031552 013702 001270      MOV      $BASE,R2      ;LOAD RK611 BASE
5674 031556 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5675 031564 012700 002000      MOV      #2000,R0      ;SET A STALL COUNT
5676 031570 005300      5$:  DEC      R0      ;LOOP UNTIL COUNT 0
5677 031572 001376      BNE      5$
5678
5679 031574 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;SET DIAGNOSTIC MODE
5680 031602 012762 054606 000004      MOV      #ECCBUF,RKBA(R2) ;LOAD BUFFER ADDRESS
5681 031610 012762 177400 000002      MOV      #-400,RKWC(R2) ; --- WORD COUNT
5682 031616 012762 010023 000000      MOV      #WRDATA!CFMT,RKCS1(R2) ; --- START COMMAND
5683 031624 012700 005136      MOV      #66.*40.+<4.*3.+2>,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5684
5685 031630 012762 000440 000026      7$:  MOV      #DMD!MCLK,RKMR1(R2)
5686 031636 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5687 031644 005300      DEC      R0
5688 031646 001370      BNE      7$
5689
5690 031650 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5691 031656 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5692
5693 031664 005037 003254      CLR      PR.BIT      ;GENERATE SYNC
5694 031670 005037 003256      CLR      M1.BIT
5695 031674 012700 000377      MOV      #255.,R0
5696
5697 031700 004737 035422      9$:  JSR      PC,RDBIT      ;SIMULATE SYNC 0 BITS
5698 031704 005300      DEC      R0
5699 031706 001374      BNE      9$
5700
5701 031710 012737 000001 003254      MCV      #1,PR.BIT      ;SIMULATE SYNC 1 BIT
5702 031716 004737 035422      JSR      PC,RDBIT
5703
5704 031722 012703 052072      MOV      #HEAD11,R3      ;SIMULATE HEADER
5705 031726 012701 000003      MOV      #3,R1      ; CYL 0 TRK 0, SEC 0
5706 031732 012304      11$:  MOV      (R3)+,R4      ;GET HEADER WORD
5707 031734 012700 000020      MOV      #16.,R0      ;LOAD BITS PER WORD

```

```

5708 031740 013737 003254 003256 13$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
5709 031746 006004 ROR R4 ;GET NEXT BIT
5710 031750 103403 BCS 15$
5711 031752 005037 003254 CLR PR.BIT
5712 031756 000403 BR 16$
5713
5714 031760 012737 000001 003254 15$: MOV #1,PR.BIT
5715 031766 004737 035422 16$: JSR PC,RDBIT ;SIMULATE NEXT BIT
5716 031772 005300 DEC R0 ;READY FOR NEXT HEADER WORD?
5717 031774 001361 BNE 13$ ;NO - GET NEXT BIT THIS WORD
5718 031776 005301 DEC R1 ;HEADER DONE?
5719 032000 001354 BNE 11$ ;NO - GET NEXT HEADER WORD
5720 032002 012700 000101 MOV #65.,R0 ;SET COUNT FOR GAP.
5721 032006 013737 003254 003256 20$: MOV PR.BIT,M1.BIT ;SIMULATE GAP
5722 032014 005037 003254 CLR PR.BIT
5723 032020 004737 035422 JSR PC,RDBIT
5724 032024 005300 DEC R0
5725 032026 001367 BNE 20$
5726 032030 012700 000400 MOV #256.,R0 ;SET COUNT FOR WRITE DATA SYNC
5727 032034 012737 047260 001310 MOV #EM327,EMW ;LOAD ERROR MESSAGE
5728 032042 005037 003252 CLR P1.BIT ;CLEAR BITS
5729 032046 005037 003254 CLR PR.BIT
5730 032052 005037 003256 CLR M1.BIT
5731 032056 005037 003260 CLR M2.BIT
5732 032062 005037 003262 CLR BITCNT ;CLEAR BIT COUNTER
5733 032066 012737 062040 003224 MOV #DMD!ECCW!MEWD!WRTGAT,E.MR1 ;SET EXPECTED MR1
5734 032074 004737 034674 22$: JSR PC,WRTBIT ;SIMULATE SYNC 0
5735 032100 104002 ERROR 2
5736 032102 005237 003262 INC BITCNT ;BUMP BIT COUNT
5737 032106 005300 DEC R0 ;LOOP UNTIL SYNC 0 WRITTEN
5738 032110 001371 BNE 22$
5739 032112 012737 000001 003252 MOV #1,P1.BIT ;SIMULATE SYNC 1
5740 032120 004737 034674 JSR PC,WRTBIT
5741 032124 104002 ERROR 2
5742 032126 005037 003266 CLR ECCHI ;INITIALIZE ECC WORDS.
5743 032132 005037 003270 CLR ECCL0
5744 032136 005037 003262 CLR BITCNT ;CLEAR BIT COUNT
5745 032142 012703 054606 MOV #ECCBUF,R3 ;SET DATA POINTER
5746 032146 012701 000377 MOV #255.,R1 ;SET DATA COUNT
5747 032152 012304 24$: MOV (R3)+,R4 ;GET DATA WORD
5748 032154 012737 050203 001310 MOV #EM338,EMW ;LOAD MESSAGE (18 BIT DATA WRITE)
5749 032162 012700 000022 MOV #18.,R0 ;LOAD BITS PER WORD
5750 032166 013737 003256 003260 25$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5751 032174 013737 003254 003256 MOV PR.BIT,M1.BIT
5752 032202 013737 003252 003254 MOV P1.BIT,PR.BIT
5753 032210 000241 CLC ;CLEAR CARRY & GET NEXT BIT
5754 032212 006004 ROR R4
5755 032214 103403 BCS 27$
5756 032216 005037 003252 CLR P1.BIT
5757 032222 000403 BR 28$
5758
5759 032224 012737 000001 003252 27$: MOV #1,P1.BIT
5760 032232 004737 034674 28$: JSR PC,WRTBIT ;SIMULATE NEXT BIT
5761 032236 104002 ERROR 2
5762 032240 016237 000032 003174 MOV RKECPT(R2),T.ECPT ;GET ECC PATTERN
5763 032246 004737 034526 JSR PC,ECCGEN ;COMPUTE EXPECTED ECC

```

```

5764 032252 023737 003174 003234      CMP      T.ECPT,E.ECPT      ;CHECK IF CORRECT
5765 032260 001401                      BEQ      29$              ;YES - SKIP
5766 032262 104164                      ERROR    164
5767 032264 005237 003262      29$:    INC      BITCNT
5768 032270 005300                      DEC      R0
5769 032272 020027 000002      CMP      R0,#2           ;1ST 16 BITS JUST DONE?
5770 032276 001403                      BEQ      31$              ;YES - SKIP
5771 032300 005700                      TST     R0                ;ELSE TEST IF WORD DONE
5772 032302 001331                      BNE     25$              ;NO - DO NEXT BIT
5773 032304 000406                      BR      32$              ;ELSE DO NEXT WORD
5774 032306 012737 050271 001310 31$:    MOV     #EM339,EMW       ;LOAD MESSAGE (BIT 16.17)
5775 032314 012704 000000      MOV     #0,R4            ;SET UPPER BITS OF WORD
5776 032320 000722                      BR      25$              ;GO DO BITS
5777
5778 032322 005301      32$:    DEC     R1                ;ALL WORDS DONE?
5779 032324 001312                      BNE     24$              ;NO - GET NEXT WORD
5780
5781 032326 012737 050203 001310      MOV     #EM338,EMW       ;SET ERROR MESSAGE
5782 032334 012304                      MOV     (R3)+,R4         ;GET LAST WORD
5783 032336 012700 000022      MOV     #18.,R0         ;SET BIT COUNT
5784 032342 013737 003256 003260 34$:    MOV     M1.BIT,M2.BIT   ;SHIFT BITS
5785 032350 013737 003254 003256      MOV     PR.BIT,M1.BIT
5786 032356 013737 003252 003254      MOV     P1.BIT,PR.BIT
5787 032364 000241                      CLC
5788 032366 006004                      ROR     R4                ;CLEAR CARRY
5789 032370 103403                      BCS     36$              ;TEST NEXT BIT
5790 032372 005037 003252      CLR     P1.BIT          ;SKIP IF A ONE
5791 032376 000403                      BR      37$              ;ELSE SET NEXT BIT FOR ZERO
5792
5793 032400 012737 000001 003252 36$:    MOV     #1,P1.BIT       ;SET NEXT BIT FOR 1
5794 032406 004737 034674      37$:    JSR     PC,WRTBIT       ;SIMULATE BIT
5795 032412 104002                      ERROR    2
5796 032414 016237 000032 003174      MOV     RKECPT(R2),T.ECPT ;GET PATTERN
5797 032422 004737 034526      JSR     PC,ECCGEN       ;GENERATE ECC PATTERN
5798 032426 023737 003174 003234      CMP     T.ECPT,E.ECPT   ;TEST IF CORRECT
5799 032434 001401                      BEQ     39$              ;YES - SKIP
5800 032436 104164                      ERROR    164            ;ELSE REPORT
5801 032440 005237 003262      39$:    INC     BITCNT          ;BUMP BIT COUNT
5802 032444 022700 000002      CMP     #2,R0           ;IF NEXT BIT IS THE LAST DATA BIT
5803 032450 001003                      BNE     41$              ;ECCW MUST BE RESET TO INDICATE
5804 032452 042737 020000 003224      BIC     #ECCW,E.MR1     ;ECC BEING WRITTEN
5805 032460 005300      41$:    DEC     R0
5806 032462 022700 000002      CMP     #2,R0           ;TEST IF 1ST 16 BITS DONE
5807 032466 001403                      BEQ     42$              ;YES - SKIP
5808 032470 005700                      TST     R0                ;ELSE TEST IF WORD DONE
5809 032472 001323                      BNE     34$              ;NO - SKIP
5810 032474 000406                      BR      45$              ;ELSE GO TO ECC PROCESSING
5811
5812 032476 012737 050271 001310 42$:    MOV     #EM339,EMW       ;SET MESSAGE FOR BITS 16 AND 17
5813 032504 012704 000000      MOV     #0,R4            ;SET UP UPPER BITS
5814 032510 000714                      BR      34$              ;GO DO LAST TWO BITS
5815 032512 012737 047677 001310 45$:    MOV     #EM333,EMW       ;MESSAGE (ECC WRITE)
5816 032520 005037 003262      CLR     BITCNT          ;CLEAR BIT COUNT
5817 032524 012703 003266      MOV     #ECCHI,R3       ;LOAD POINTER TO ECC WORDS
5818 032530 012701 000002      MOV     #2,R1            ;SET FOR 2 WORDS
5819 032534 012304      43$:    MOV     (R3)+,R4         ;GET ECC WORD
    
```

```

5820 032536 012700 000020          MOV      #16.,R0          ;SET BIT COUNT
5821 032542 013737 003256 003260 44$: MOV      M1.BIT,M2.BIT    ;SHIFT BITS
5822 032550 013737 003254 003256  MOV      PR.BIT,M1.BIT
5823 032556 013737 003252 003254  MOV      P1.BIT,PR.BIT
5824 032564 006004          ROR      R4              ;LOAD NEXT BIT
5825 032566 103403          BCS     46$
5826 032570 005037 003252          CLR     P1.BIT
5827 032574 000403          BR      47$
5828
5829 032576 012737 000001 003252 46$: MOV      #1,P1.BIT
5830 032604 004737 034674 47$: JSR     PC,WRTBIT    ;SIMULATE NEXT BIT
5831 032610 104002          ERROR   2
5832 032612 022700 000002          CMP     #2,R0          ;IF THIS BIT IS THE LAST ECC BIT
5833 032616 001006          BNE     49$            ;ECCW MUST BE SET TO INDICATE
5834 032620 022701 000001          CMP     #1,R1          ;ECC IS DONE
5835 032624 001003          BNE     49$
5836 032626 052737 020000 003224 49$: BIS     #ECCW,E.MR1
5837 032634 005237 003262          INC     BITCNT         ;BUMP BIT COUNT
5838 032640 005300          DEC     R0             ;TEST IF LAST BIT THIS WORD IS DONE
5839 032642 001337          BNE     44$           ;NO - LOOP
5840 032644 005301          DEC     R1             ;TEST IF BOTH WORDS WRITTEN
5841 032646 001332          BNE     43$           ;NO - LOOP
5842 032650 012737 047735 001310  MOV     #EM334,EMW     ;LOAD MESSAGE (POSTAMBLE)
5843 032656 005037 003262          CLR     BITCNT        ;CLEAR BIT COUNT
5844 032662 012700 000017          MOV     #15.,R0       ;SET GAP COUNT
5845 032666 013737 003256 003260 51$: MOV     M1.BIT,M2.BIT  ;SHIFT REST OF BITS
5846 032674 013737 003254 003256  MOV     PR.BIT,M1.BIT
5847 032702 013737 003252 003254  MOV     P1.BIT,PR.BIT
5848 032710 005037 003252          CLR     P1.BIT        ;CLEAR NEXT BIT
5849 032714 004737 034674          JSR     PC,WRTBIT     ;SIMULATE BIT
5850 032720 104002          ERROR   2
5851 032722 005237 003262          INC     BITCNT        ;BUMP BIT COUNT
5852 032726 005300          DEC     R0             ;GAP WRITTEN?
5853 032730 001356          BNE     51$           ;NO - LOOP
5854 032732 012737 000001 003234  MOV     #1,E.ECPT     ;SET EXPECTED PATTERN
5855 032740 016237 000032 003174  MOV     RKECPT(R2),T.ECPT ;GET '611 PATTERN
5856 032746 023737 003174 003234  CMP     T.ECPT,E.ECPT ;TEST IF EQUAL
5857 032754 001401          BEQ     56$           ;YES - SKIP
5858 032756 104163          ERROR   163          ;ELSE REPORT
5859 032760          56$:
5860          .SBTTL  END OF PASS ROUTINE
5861
5862          ;*****
5863          ;*INCREMENT THE PASS NUMBER ($PASS)
5864          ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
5865          ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
5866          ;*IF THERES A MONITOR GO TO IT
5867          ;*IF THERE ISN'T JUMP TO TST1
5868
5869          $EOP:
5870          032760 000004          SCOPE
5871 032762 005037 001102          CLR     $STNM         ;;ZERO THE TEST NUMBER
5872 032766 005037 001200          CLR     $TIMES       ;;ZERO THE NUMBER OF ITERATIONS
5873 032772 005237 001222          INC     $PASS        ;;INCREMENT THE PASS NUMBER
5874 032776 042737 100000 001222  BIC     #100000,$PASS ;DON'T ALLOW A NEG. NUMBER
5875 033004 005327          DEC     (PC)+        ;;LOOP?

```

```

5876 033006 000001
5877 033010 003063
5878 033012 012737
5879 033014 000001
5880 033016 033006
5881 033020 104401 033026
5882 033024 000407
5883
5884 033044
5885 033044 013746 001222
5886
5887 033050 104405
5888 033052 104401 033060
5889 033056 000421
5890
5891 033122
5892 033122 013746 001112
5893
5894 033126 104405
5895 033130 104401 001211
5896 033134 005037 001112
5897 033140 013700 000042
5898 033144 001405
5899 033146 000005
5900 033150 004710
5901 033152 000240
5902 033154 000240
5903 033156 000240
5904 033160
5905 033160 000137
5906 033162 004312
5907 033164 377 377 000
5908 033170
5909
5910
5911
5912
5913
5914
5915
5916 033170 013702 001270 000000
5917 033174 012762 100000
5918 033202 104401 042221
5919 033206 012706 001100
5920 033212 005037 001202
5921 033216 105037 001103
5922 033222 005737 000042
5923 033226 001404
5924 033230 005037 033006
5925 033234 000137 032760
5926
5927 033240 000000
5928 033242 000137 003342
5929
5930
5931

$EOPCT: .WORD 1
          BGT $DOAGN ;;YES
          MOV (PC)+,@(PC)+ ;;RESTORE COUNTER
$ENDCT: .WORD 1
          $EOPCT
          TYPE ,65$ ;;TYPE ASCIZ STRING
          BR 64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <12><15>/END PASS #/
64$:
          MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
          TYPDS ;;TYPE PASS NUMBER
          TYPE ,67$ ;;GO TYPE--DECIMAL ASCII WITH SIGN
          BR 66$ ;;TYPE ASCIZ STRING
;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
66$:
          MOV $ERTTL,-(SP) ;;SAVE $ERTTL FOR TYPEOUT
          TYPDS ;;TOTAL NUMBER OF ERRORS
          TYPE , $CRLF ;;GO TYPE--DECIMAL ASCII WITH SIGN
          CLR $ERTTL ;;TYPE CARRIAGE RETURN, LINE FEED
          MOV @#42,R0 ;;CLEAR ERROR TOTAL
          BEQ $DOAGN ;;GET MONITOR ADDRESS
          RESET ;;BRANCH IF NO MONITOR
          JSR PC,(R0) ;;CLEAR THE WORLD
          NOP ;;GO TO MONITOR
          NOP ;;SAVE ROOM
          NOP ;;FOR
          ;;ACT11
          JMP @(PC)+ ;;RETURN
$RTNAD: .WORD TST1
$ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
          .EVEN

.SBTTL CONTROLLED HALT ROUTINE
;* THIS ROUTINE IS ENTERED WHEN A ^C IS ENTERED IN THE KEYBOARD.
;* IF NO MONITOR IS PRESENT THE PROGRAM HALTS ELSE THE PROGRAM
;* IS FORCED TO LAST PASS AND JUMPS TO END OF PASS.
CTRHLT: MOV $BASE,R2 ;SET '611 BASE
          MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
          TYPE ,OPR005 ;TYPE HALT MESSAGE
          MOV #STACK,SP ;CLEAR STACK
          CLR $ESCAPE ;CLEAR ESCAPE
          CLRB $ERFLG ;CLEAR ERROR FLAG
          TST 42 ;TEST IF STAND ALONE
          BEQ 1$ ;YES - SKIP
          CLR $EOPCT ;ZERO PASS COUNT
          JMP $EOP ;GO TO END OF PASS
1$: HALT ;HALT PROGRAM
          JMP START1 ;GO TO RESTART IF CONTINUE

.SBTTL SCOPE HANDLER ROUTINE

```

```
5932 ::*****
5933 ::*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
5934 ::*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
5935 ::*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
5936 ::*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
5937 ::*SW14=1 LOOP ON TEST
5938 ::*SW11=1 INHIBIT ITERATIONS
5939 ::*SW09=1 LOOP ON ERROR
5940 ::*SW08=1 LOOP ON TEST IN SWR<7:0>
5941 ::*CALL
5942 ::* SCOPE ;;SCOPE=IOT
5943
5944 $SCOPE:
5945 033246 104407
5946 033250 032777 040000 145662 1$: CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
5947 033256 001131 BIT #BIT14,@SWR ;;LOOP ON PRESENT TEST?
5948 BNE $OVER ;;YES IF SW14=1
5949 033260 000416 :*****START OF CODE FOR THE XOR TESTER*****
5950 $XTSTR: BR 6$ ;;IF RUNNING ON THE "XOR" TESTER CHANGE
5951 033262 013746 000004 MOV @#ERRVEC,-(SP) ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
5952 033266 012737 033306 000004 MOV #5,@#ERRVEC ;;SAVE THE CONTENTS OF THE ERROR VECTOR
5953 033274 005737 177060 TST @#177060 ;;SET FOR TIMEOUT
5954 033300 012637 000004 MOV (SP)+,@#ERRVEC ;;TIME OUT ON XOR?
5955 033304 000500 BR $SVLAD ;;RESTORE THE ERROR VECTOR
5956 033306 022626 5$: CMP (SP)+,(SP)+ ;;GO TO THE NEXT TEST
5957 033310 012637 000004 MOV (SP)+,@#ERRVEC ;;CLEAR THE STACK AFTER A TIME OUT
5958 033314 000440 BR 7$ ;;RESTORE THE ERROR VECTOR
5959 033316 6$:;*****END OF CODE FOR THE XOR TESTER***** ;;LOOP ON THE PRESENT TEST
5960 033316 032777 000400 145614 BIT #BIT08,@SWR ;;LOOP ON SPEC. TEST?
5961 033324 001421 BEQ 2$ ;;BR IF NO
5962 033326 005046 CLR -(SP) ;;CLEAR A TEMP. LOCATION
5963 033330 117716 145604 MOVB @SWR,(SP) ;;PICKUP THE DESIRED TEST NUMBER
5964 033334 001414 BEQ 8$ ;;BRANCH IF BAD TEST NUMBER IN SWR
5965 033336 022716 000047 CMP #47,(SP) ;;CHECK THE NUMBER IN THE SWR
5966 033342 002411 BLT 8$ ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
5967 033344 011637 001102 MOV (SP),$TSTNM ;;UPDATE THE TEST NUMBER
5968 033350 005316 DEC (SP) ;;BACKUP BY ONE
5969 033352 006316 ASL (SP) ;;SCALE THE TEST NUMBER AS AN INDEX
5970 033354 062716 033560 ADD #SW08TBL,(SP) ;;FORM THE ADDRESS OF TEST POINTER
5971 033360 013637 001106 MOV @($)+,$LPADR ;;SET LOOP ADDRESS TO DESIRED TEST
5972 033364 000466 BR $OVER ;;GO LOOP ON THE TEST
5973 033366 005726 8$: TST (SP)+ ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
5974 033370 105737 001103 2$: TSTB $ERFLG ;;HAS AN ERROR OCCURRED?
5975 033374 001421 BEQ 3$ ;;BR IF NO
5976 033376 123737 001115 001103 CMPB $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
5977 033404 101015 BHI 3$ ;;BR IF NO
5978 033406 032777 001000 145524 BIT #BIT09,@SWR ;;LOOP ON ERROR?
5979 033414 001404 BEQ 4$ ;;BR IF NO
5980 033416 013737 001110 001106 7$: MOV $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
5981 033424 000446 BR $OVER
5982 033426 105037 001103 4$: CLRB $ERFLG ;;ZERO THE ERROR FLAG
5983 033432 005037 001200 CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
5984 033436 000415 BR 1$ ;;ESCAPE TO THE NEXT TEST
5985 033440 032777 004000 145472 3$: BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?
5986 033446 001011 BNE 1$ ;;BR IF YES
5987 033450 005737 001222 TST $PASS ;;IF FIRST PASS OF PROGRAM
```

```
5988 033454 001406          BEQ      1$          ;;      INHIBIT ITERATIONS
5989 033456 005237 001104    INC      $ICNT      ;;      INCREMENT ITERATION COUNT
5990 033462 023737 001200 001104    CMP      $TIMES,$ICNT ;;      CHECK THE NUMBER OF ITERATIONS MADE
5991 033470 002024          BGE      $OVER      ;;      BR IF MORE ITERATION REQUIRED
5992 033472 012737 000001 001104 1$:      MOV      #1,$ICNT  ;;      REINITIALIZE THE ITERATION COUNTÉR
5993 033500 013737 033556 001200    MOV      $MXCNT,$TIMES ;;      SET NUMBER OF ITERATIONS TO DO
5994 033506 105237 001102          $SVLAD: INCB     $TSTNM  ;;      COUNT TEST NUMBERS
5995 033512 113737 001102 001220    MOVB    $TSTNM,$TESTN ;;      SET TEST NUMBER IN APT MAILBOX
5996 033520 011637 001106          MOV      (SP),$LPADR  ;;      SAVE SCOPE LOOP ADDRESS
5997 033524 011637 001110          MOV      (SP),$LPERR  ;;      SAVE ERROR LOOP ADDRESS
5998 033530 005037 001202          CLR      $ESCAPE     ;;      CLEAR THE ESCAPE FROM ERROR ADDRESS
5999 033534 112737 000001 001115    MOVB    #1,$ERMAX    ;;      ONLY ALLOW ONE(1) ERROR ON NEXT TEST
6000 033542 013777 001102 145372 $OVER:   MOV      $TSTNM,@DISPLAY ;;      DISPLAY TEST NUMBER
6001 033550 013716 001106          MOV      $LPADR,(SP) ;;      FUDGE RETURN ADDRESS
6002 033554 000002          RTI              ;;      FIXES PS
6003 033556 003720          $MXCNT: 2000.    ;;      MAX. NUMBER OF ITERATIONS
6004 033560          $SWO8TBL:
6005 033560 004314          .WORD   TST1+2      ;;      STARTING ADDRESS OF TEST 1
6006 033562 004546          .WORD   TST2+2      ;;      STARTING ADDRESS OF TEST 2
6007 033564 005000          .WORD   TST3+2      ;;      STARTING ADDRESS OF TEST 3
6008 033566 005232          .WORD   TST4+2      ;;      STARTING ADDRESS OF TEST 4
6009 033570 005462          .WORD   TST5+2      ;;      STARTING ADDRESS OF TEST 5
6010 033572 005712          .WORD   TST6+2      ;;      STARTING ADDRESS OF TEST 6
6011 033574 006142          .WORD   TST7+2      ;;      STARTING ADDRESS OF TEST 7
6012 033576 006426          .WORD   TST10+2     ;;      STARTING ADDRESS OF TEST 10
6013 033600 006676          .WORD   TST11+2     ;;      STARTING ADDRESS OF TEST 11
6014 033602 007146          .WORD   TST12+2     ;;      STARTING ADDRESS OF TEST 12
6015 033604 007422          .WORD   TST13+2     ;;      STARTING ADDRESS OF TEST 13
6016 033606 010072          .WORD   TST14+2     ;;      STARTING ADDRESS OF TEST 14
6017 033610 010516          .WORD   TST15+2     ;;      STARTING ADDRESS OF TEST 15
6018 033612 011056          .WORD   TST16+2     ;;      STARTING ADDRESS OF TEST 16
6019 033614 011450          .WORD   TST17+2     ;;      STARTING ADDRESS OF TEST 17
6020 033616 012044          .WORD   TST20+2     ;;      STARTING ADDRESS OF TEST 20
6021 033620 012440          .WORD   TST21+2     ;;      STARTING ADDRESS OF TEST 21
6022 033622 013034          .WORD   TST22+2     ;;      STARTING ADDRESS OF TEST 22
6023 033624 013430          .WORD   TST23+2     ;;      STARTING ADDRESS OF TEST 23
6024 033626 014216          .WORD   TST24+2     ;;      STARTING ADDRESS OF TEST 24
6025 033630 014706          .WORD   TST25+2     ;;      STARTING ADDRESS OF TEST 25
6026 033632 015302          .WORD   TST26+2     ;;      STARTING ADDRESS OF TEST 26
6027 033634 015754          .WORD   TST27+2     ;;      STARTING ADDRESS OF TEST 27
6028 033636 016426          .WORD   TST30+2     ;;      STARTING ADDRESS OF TEST 30
6029 033640 017100          .WORD   TST31+2     ;;      STARTING ADDRESS OF TEST 31
6030 033642 017552          .WORD   TST32+2     ;;      STARTING ADDRESS OF TEST 32
6031 033644 020340          .WORD   TST33+2     ;;      STARTING ADDRESS OF TEST 33
6032 033646 021126          .WORD   TST34+2     ;;      STARTING ADDRESS OF TEST 34
6033 033650 021714          .WORD   TST35+2     ;;      STARTING ADDRESS OF TEST 35
6034 033652 022404          .WORD   TST36+2     ;;      STARTING ADDRESS OF TEST 36
6035 033654 023074          .WORD   TST37+2     ;;      STARTING ADDRESS OF TEST 37
6036 033656 023564          .WORD   TST40+2     ;;      STARTING ADDRESS OF TEST 40
6037 033660 024254          .WORD   TST41+2     ;;      STARTING ADDRESS OF TEST 41
6038 033662 024770          .WORD   TST42+2     ;;      STARTING ADDRESS OF TEST 42
6039 033664 025514          .WORD   TST43+2     ;;      STARTING ADDRESS OF TEST 43
6040 033666 026240          .WORD   TST44+2     ;;      STARTING ADDRESS OF TEST 44
6041 033670 027476          .WORD   TST45+2     ;;      STARTING ADDRESS OF TEST 45
6042 033672 030746          .WORD   TST46+2     ;;      STARTING ADDRESS OF TEST 46
6043 033674 031544          .WORD   TST47+2     ;;      STARTING ADDRESS OF TEST 47
```

```
6044      ;:*****
6045      .SBTTL LOOP ON INTERNAL ERROR
6046
6047 033676 032777 001000 145234 SCOP1$: BIT    #SW9,@SWR    ;CHECK IF LOOP ON ERROR
6048 033704 001405          BEQ     5$          ;NO RETURN
6049 033706 105737 001103          TSTB   $ERFLG    ;CHECK IF ERROR OCCURED
6050 033712 001402          BEQ     5$          ;NO, RETURN
6051 033714 013716 001110          MOV    $LPERR,(SP) ;GO BACK TO BEGINNING OF LOOP
6052 033720 000002          5$:    RTI          ;RETURN
6053
6054      ;:*****
6055      .SBTTL TYPE ERROR ROUTINE
6056      ;*ENTRY JSR PC,TYPERR
6057      ;*RETURN RTS PC
6058      ;*
6059      ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
6060      ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
6061      ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
6062      ;*THE ERROR.
6063      ;:*****
6064 033722 104413          TYPERR: SAVREG
6065 033724 113737 001102 001220      MOVB   $STNM,$TESTN ;GET TEST NUMBER
6066 033732 042737 177400 001220      BIC   #177400,$TESTN ;CLEAR UNUSED BITS
6067 033740 113700 001114          MOVB   $ITEMB,R0    ;ENTER ERROR NUMBER
6068 033744 042700 177400          BIC   #177400,R0    ;CLEAR UNUSED BITS
6069 033750 005300          DEC   R0           ;FORM INDEX FOR ERROR TABLE
6070 033752 006300          ASL   R0
6071 033754 006300          ASL   R0
6072 033756 006300          ASL   R0
6073 033760 062700 001300          1$:  ADD   #$ERRTB,R0    ;FORM ADDRESS OF ERROR ENTRY
6074 033764 012037 034000          MOV   (R0)+,2$     ;GET EM POINTER
6075 033770 001404          BEQ   3$           ;BRANCH IF THERE ISN'T ONE
6076 033772 104401 001211          TYPE , $CRLF      ;TYPE CARRIAGE RETURN LINE FEED
6077 033776 104401          TYPE          ;TYPE ERROR MESSAGE (EM)
6078 034000 000000          2$:  .WORD 0        ;EM POINTER GOES HERE
6079 034002 012037 034016          3$:  MOV   (R0)+,4$     ;GET DH POINTER
6080 034006 001404          BEQ   5$           ;BRANCH IF THERE ISN'T ONE
6081 034010 104401 001211          TYPE , $CRLF      ;TYPE CR-LF
6082 034014 104401          TYPE          ;TYPE DATA HEADER
6083 034016 000000          4$:  .WORD 0        ;DH POINTER GOES HERE
6084 034020 012001          5$:  MOV   (R0)+,R1    ;GET DT POINTER
6085 034022 001445          BEQ   20$          ;BRANCH IF THERE ARE NONE
6086 034024 005004          CLR   R4          ;RESET INDENT SWITCH
6087 034026 012000          MOV   (R0)+,R0    ;GET DF POINTER
6088 034030 012002          MOV   (R0)+,R2    ;STORE NUMBER OF DH'S
6089 034032 104401 001211          TYPE , $CRLF
6090 034036 112003          10$: MOVB  (R0)+,R3     ;GET & STORE NUMBER OF DATA WORDS
6091 034040 105720          TSTB (R0)+        ;BUMP PAST FORMAT WORD
6092 034042 005703          TST  R3          ;TEST IF ANY DATA FOR THIS HEADER
6093 034044 001416          BEQ   14$          ;NO - SKIP DATA PRINT
6094 034046 005704          TST  R4          ;CHECK IF INDENT WORDS
6095 034050 001004          BNE  12$          ;YES, GO INDENT
6096 034052 013146          11$: MOV   @ (R1)+,-(SP) ;PUT FIRST DATA WORD ON STACK
6097 034054 104402          TYPOC          ;TYPE IT
6098 034056 005303          DEC  R3          ;MORE DATA WORDS
6099 034060 001403          BEQ  13$          ;NO-BRANCH
```



```

6100 034062 104401 042341      12$:  TYPE      SPACE2      ;TYPE SEPARATORS
6101 034066 000771              BR          11$          ;LOOP
6102 034070 104401 001211      13$:  TYPE      ,SCRLF      ;TYPE <CR><LF>
6103 034074 005710              TST         (R0)        ;CHECK IF NEXT HEADER AVAILABLE
6104 034076 001401              BEQ         14$          ;NO, DO NOT CHANGE INDENT
6105 034100 005104              COM         R4           ;CHANGE INDENT
6106 034102 005302              14$:  DEC         R2           ;MORE DH'S?
6107 034104 003414              BLE         20$          ;NO-BRANCH
6108 034106 012037 034126      15$:  MOV         (R0)+,18$    ;GET NEXT DH POINTER
6109 034112 001751              BEQ         10$          ;IF NO HEADER GET DATA
6110 034114 005704              TST         R4           ;INDENT?
6111 034116 001402              BEQ         17$          ;NO-BRANCH
6112 034120 104401 042341              TYPE      ,SPACE2      ;INDENT
6113 034124 104401              17$:  TYPE      ;TYPE DH
6114 034126 000000              18$:  .WORD     0          ;DH POINTER GOES HERE
6115 034130 104401 001211              TYPE      ,SCRLF      ;LOOP
6116 034134 000740              BR          10$
6117 034136 104414              20$:  RESREG
6118 034140 005237 003250              INC         ERRCNT      ;INCREMENT ERROR COUNT
6119 034144 032777 010000 144766              BIT         #SW12,@SWR  ;CHECK IF ABORT AFTER 20 ERRORS
6120 034152 001421              BEQ         25$          ;NO, RETURN
6121 034154 022737 000024 003250              CMP         #20.,ERRCNT ;CHECK IF EROR THRESHOLD EXCEEDED
6122 034162 103015              BHIS        25$          ;NO, RETURN
6123 034164 104401. 042344              TYPE      ,ABORT      ;TYPE 'PROGRAM HAS BEEN ABORTED BECAUSE
6124                                ; ERROR THRESHOLD EXCEEDED'
6125 034170 005737 000042              TST         42          ;CHECK IF IN CHAIN MODE
6126 034174 001407              BEQ         30$          ;NO, HALT
6127 034176 012737 000001 033006              MOV         #1,$EOPCT   ;FORCE END OF PASS COUNT TO ONE FOR ABORT
6128 034204 012706 001100              MOV         #STACK,SP  ;INITIALIZE STACK
6129 034210 000137 032760              JMP         $EOP        ;BRING IN NEXT PROGRAM IN CHAIN
6130 034214 000000              30$:  HALT
6131 034216 000207              25$:  RTS          PC
6132
6133                                .SBTTL  GENERATE HEADERS FOR SECTOR, TRACK, AND CYLINDER INCREMENTS
6134
6135 034220 013737 003300 003302  INCHDR: MOV         CYLN,HEADER ;LOAD HEADER WORD 1
6136 034226 005037 003304              CLR         HEADER+2    ;CALCULATE HEADER WORD 2
6137 034232 113737 003277 003305              MOVB        TRACK,HEADER+3
6138 034240 006237 003304              ASR         HEADER+2
6139 034244 006237 003304              ASR         HEADER+2
6140 034250 006237 003304              ASR         HEADER+2
6141 034254 153737 003276 003304              BISB        SECTOR,HEADER+2
6142 034262 052737 140000 003304              BIS         #140000,HEADER+2
6143 034270 013746 003302              MOV         HEADER,-(SP) ;GENERATE XOR
6144 034274 013737 003304 003306              MOV         HEADER+2,HEADER+4
6145 034302 043737 003302 003306              BIC         HEADER,HEADER+4
6146 034310 043716 003304              BIC         HEADER+2,(SP)
6147 034314 052637 003306              BIS         (SP)+,HEADER+4
6148 034320 013737 003300 003220              MOV         CYLN,E.DCYL ;INCREMENT DISK ADDRESS
6149 034326 013737 003276 003206              MOV         SECTOR,E.DA
6150 034334 105237 003206              INCB        E.DA
6151 034340 122737 000026 003206              CMPB        #26,E.DA
6152 034346 001014              BNE         10$
6153 034350 105037 003206              CLRB        E.DA
6154 034354 105237 003207              INCB        E.DA+1
6155 034360 122737 000003 003207              CMPB        #3,E.DA+1

```

```

6156 034366 001004          BNE      10$
6157 034370 005037 003206   CLR      E.DA
6158 034374 005237 003220   INC      E.DCYL
6159 034400 000207          RTS      PC          ;RETURN
6160
6161          .SBTTL  CALCULATE EXPECTED HEADER
6162
6163 034402 005003          CALHDR: CLR      R3          ;CLEAR EXPECTED FIRST WORD
6164 034404 013701 003220   MOV      E.DCYL,R1       ;STORE CYLINDER
6165 034410 001406          BEQ      5$              ;CHECK IF ZERO
6166 034412 012700 000020   MOV      #16.,R0        ;LOAD SHIFT COUNT
6167 034416 006101          1$:    ROL      R1          ;CALCULATE FIRST WORD
6168 034420 006003          ROR      R3
6169 034422 005300          DEC      R0
6170 034424 001374          BNE      1$
6171 034426 010337 003226   5$:    MOV      R3,E.MR2     ;LOAD FIRST WORD
6172 034432 013746 003206   MOV      E.DA,-(SP)      ;GET TRACK AND SECTOR
6173 034436 001410          BEQ      7$              ;CHECK IF TRACK = 0 AND SECTOR = 0
6174 034440 042716 000377   BIC      #377,(SP)       ;CLEAR SECTOR BITS
6175 034444 001403          BEQ      6$              ;CHECK TRACK = 0
6176 034446 006216          ASR      (SP)           ;SHIFT HEAD 3 BITS RIGHT
6177 034450 006216          ASR      (SP)
6178 034452 006216          ASR      (SP)
6179 034454 153716 003206   6$:    BISB     E.DA,(SP)   ;OR IN SECTOR BITS
6180 034460 012601          7$:    MOV      (SP)+,R1
6181 034462 032737 010000 003200 BIT      #CFMT,E.CS1     ;CHECK FORMAT = 24 SECTOR
6182 034470 001402          BEQ      8$              ;NO, CALCULATE SECOND WORD
6183 034472 052701 001000   BIS      #BIT9,R1        ;SET FORMAT BIT
6184 034476 005003          8$:    CLR      R3          ;CLEAR EXPECTED SECOND WORD
6185 034500 005701          TST      R1              ;CHECK IF WORD = 0
6186 034502 001406          BEQ      15$             ;YES, RETURN
6187 034504 012700 000020   MOV      #16.,R0        ;LOAD SHIFT COUNT
6188 034510 006101          10$:   ROL      R1
6189 034512 006003          ROR      R3
6190 034514 005300          DEC      R0
6191 034516 001374          BNE      10$
6192 034520 010337 003230   15$:   MOV      R3,E.MR3     ;LOAD SECOND WORD
6193 034524 000207          RTS      PC          ;RETURN
6194
6195          ;:*****
6196          .SBTTL  GENERATE ECC WORD
6197
6198 034526 005737 003252          ECCGEN: TST      P1.BIT   ;CHECK IF 1
6199 034532 001410          BEQ      3$              ;NO, CHECK IF BIT 31 = 1
6200 034534 032737 000001 003266 BIT      #1,ECCHI        ;NO, CHECK IF BIT 31 = 1
6201 034542 001410          BEQ      5$              ;NO, SET ECC XORED BIT
6202 034544 005037 003272          1$:    CLR      ECCXOR     ;CLEAR ECC XORED BIT
6203 034550 000241          CLC                       ;CLEAR CARRY FLOP
6204 034552 000410          BR       7$              ;SHIFT IN ECC BIT
6205
6206 034554 032737 000001 003266 3$:    BIT      #1,ECCHI        ;CHECK IF BIT 31 = 1
6207 034562 001770          BEQ      1$              ;NO, CLEAR ECC XORED BIT
6208 034564 012737 000001 003272 5$:    MOV      #1,ECCXOR     ;SET ECC XOR
6209 034572 000261          SEC                       ;SET CARRY FLOP
6210 034574 006037 003270          7$:    ROR      ECCLO
6211 034600 006037 003266          ROR      ECCHI

```

```

6212 034604 005737 003272      TST      ECCXOR      ;CHECK IF XOR NEEDED
6213 034610 001422              BEQ      10$         ;NO, RETURN
6214 034612 012746 020020      MOV      #BIT13:BIT4,-(SP) ;DO XOR OF ECC BITS
6215 034616 043716 003270      BIC      ECCLO,(SP)   ; 0, 2, 11, 21, 23
6216 034622 042737 020020 003270      BIC      #BIT13:BIT4,ECCLO
6217 034630 052637 003270      BIS      (SP)+,ECCLO
6218 034634 012746 002400      MOV      #BIT10:BIT8,-(SP)
6219 034640 043716 003266      BIC      ECCHI,(SP)
6220 034644 042737 002400 003266      BIC      #BIT10:BIT8,ECCHI
6221 034652 052637 003266      BIS      (SP)+,ECCHI
6222 034656 013737 003266 003234 10$:  MOV      ECCHI,E.ECPT   ;STORE ECC PATTERN
6223 034664 042737 174000 003234      BIC      #174000,E.ECPT ;MASK UNSEEN BITS
6224 034672 000207              RTS      PC           ;RETURN
6225
6226      ;*****
6227      .SBTTL SIMULATE ONE BIT WRITE IN MAINTENCE MODE
6228 034674 052737 042040 003224 WRTBIT: BIS      #DMD!MEWD!WRTGAT,E.MR1 ;INITIALIZE
6229 034702 042737 114000 003224      BIC      #RDGATE!PCA!PCD,E.MR1 ; EXPECTED MR1
6230 034710 005737 003254      TST      PR.BIT      ;CHECK IF ONE
6231 034714 001122              BNE      20$         ;YES, SIMULATE A ONE
6232 034716 005737 003256      TST      M1.BIT      ;CHECK IF PREVIOUS ONE
6233 034722 001023              BNE      10$         ;YES, NO TRANSITION
6234 034724 042737 002000 003224      BIC      #MEWD,E.MR1  ;INDICATE TRANSITION
6235 034732 005737 003252      TST      P1.BIT      ;CHECK IF NEXT BIT = 1
6236 034736 001007              BNE      5$          ;YES, CHECK FOR PRECOMP ADVANCE
6237 034740 005737 003260      TST      M2.BIT      ;CHECK FOR PRECOMP. DELAY
6238 034744 001412              BEQ      10$         ;NO, CLOCK IN ZERO
6239 034746 052737 010000 003224      BIS      #PCD,E.MR1  ;SET PRECOMP. DELAY
6240 034754 000406              BR       10$         ;CLOCK IN ZERO
6241
6242 034756 005737 003260      5$:  TST      M2.BIT      ;CHECK FOR PRECOMP. ADVANCE
6243 034762 001003              BNE      10$         ;CLOCK IN ZERO
6244 034764 052737 004000 003224      BIS      #PCA,E.MR1  ;SET PRECOMPENSATION ADVANCE
6245 034772 012762 000440 000026 10$:  MOV      #DMD!MCLK,RKMR1(R2) ;CLOCK IN DATA BIT
6246 035000 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6247 035006 016237 000026 003164      MOV      RKMR1(R2),T.MR1 ;STORE MR1
6248 035014 023737 003164 003224      CMP      T.MR1,E.MR1  ;CHECK IF MR1 CORRECT
6249 035022 001416              BEQ      15$         ;YES, CLOCK IN REST OF BIT
6250 035024 012737 035044 001202      MOV      #13$, $ESCAPE ;LOAD ESCAPE
6251 035032 012737 051363 001312      MOV      #EMW2,EMW+2  ;LOAD ERROR MESSAGE
6252 035040 011646              MOV      (SP),-(SP)  ;STORE RETURN
6253 035042 000207              RTS      PC           ;REPORT ERROR
6254 035044 032777 001000 144066 13$:  BIT      #SW9,@SWR    ;CHECK IF LOOP ON ERROR
6255 035052 001402              BEQ      15$         ;NO, CONTINUE
6256 035054 000137 035406              JMP      63$         ;GO LOOP ON ERROR
6257
6258 035060 052737 002000 003224 15$:  BIS      #MEWD,E.MR1  ;RESET TRANSITION INDICATION
6259 035066 012762 000440 000026      MOV      #DMD!MCLK,RKMR1(R2) ;CLOCK IN DATA BIT
6260 035074 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6261 035102 016237 000026 003164      MOV      RKMR1(R2),T.MR1 ;STORE MR1
6262 035110 023737 003164 003224      CMP      T.MR1,E.MR1  ;CHECK IF MR1 CORRECT
6263 035116 001414              BEQ      18$         ;YES, RETURN
6264 035120 012737 035140 001202      MOV      #17$, $ESCAPE ;LOAD ESCAPE
6265 035126 012737 051453 001312      MOV      #EMW4,EMW+2  ;LOAD ERROR MESSAGE
6266 035134 011646              MOV      (SP),-(SP)  ;SAVE RETURN
6267 035136 000207              RTS      PC           ;REPORT ERROR

```

```
6268
6269 035140 032777 001000 143772 17$: BIT #SW9,@SWR ;CHECK IF LOOP ON ERROR
6270 035146 001117 BNE 63$ ;YES, LOOP ERROR
6271 035150 005037 001202 18$: CLR $ESCAPE ;CLEAR ESCAPE
6272 035154 062716 000002 ADD #2,(SP) ;ADJUST ESCAPE
6273 035160 000207 RTS PC ;RETURN
6274
6275
6276 035162 005737 003252 20$: TST P1.BIT ;CHECK IF NEXT BIT A ONE
6277 035166 001007 BNE 30$ ;YES, CHECK FOR PRECOMP. DELAY
6278 035170 005737 003256 TST M1.BIT ;CHECK FOR PRECOMP. ADVANCE
6279 035174 001412 BEQ 40$ ;NO, CHECK MR1
6280 035176 052737 004000 003224 BIS #PCA,E.MR1 ;SET PRECOMP ADVANCE
6281 035204 000406 BR 40$ ;CHECK MR1
6282
6283 035206 005737 003256 30$: TST M1.BIT ;CHECK FOR PRECOMP. DELAY
6284 035212 001003 BNE 40$ ;NO, CHECK MR1
6285 035214 052737 010000 003224 BIS #PCD,E.MR1 ;SET PRECOMP. DELAY
6286 035222 012762 000440 000026 40$: MOV #DMD!MCLK,RKMR1(R2) ;CLOCK IN DATA BIT
6287 035230 012762 000040 000026 MOV #DMD,RKMR1(R2)
6288 035236 016237 000026 003164 MOV RKMR1(R2),T.MR1 ;STORE MR1
6289 035244 023737 003164 003224 CMP T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
6290 035252 001414 BEQ 45$ ;YES, CLOCK IN REST OF BIT
6291 035254 012737 035274 001202 MOV #42,$ESCAPE ;LOAD ESCAPE
6292 035262 012737 051363 001312 MOV #EMW2,EMW+2 ;LOAD ERROR MESSAGE
6293 035270 011646 MOV (SP),-(SP) ;STORE RETURN
6294 035272 000207 RTS PC ;REPORT ERROR
6295
6296 035274 032777 001000 143636 42$: BIT #SW9,@SWR ;CHECK IF LOOP ON ERROR
6297 035302 001041 BNE 63$ ;YES, LOOP ERROR
6298 035304 042737 002000 003224 45$: BIC #MEWD,E.MR1 ;INDICATE A TRANSITION
6299 035312 012762 000440 000026 MOV #DMD!MCLK,RKMR1(R2) ;CLOCK TRANSITION
6300 035320 012762 000040 000026 MOV #DMD,RKMR1(R2)
6301 035326 016237 000026 003164 MOV RKMR1(R2),T.MR1 ;STORE MR1
6302 035334 023737 003164 003224 CMP T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
6303 035342 001414 BEQ 50$ ;YES, RETURN
6304 035344 012737 035364 001202 MOV #47,$ESCAPE ;LOAD ESCAPE
6305 035352 012737 051453 001312 MOV #EMW4,EMW+2 ;LOAD ERROR MESSAGE
6306 035360 011646 MOV (SP),-(SP) ;SAVE RETURN
6307 035362 000207 RTS PC ;REPORT ERROR
6308
6309 035364 032777 001000 143546 47$: BIT #SW9,@SWR ;CHECK IF LOOP ON ERROR
6310 035372 001005 BNE 63$ ;YES, REPORT ERROR
6311 035374 005037 001202 50$: CLR $ESCAPE ;LOAD ESCAPE
6312 035400 062716 000002 ADD #2,(SP) ;ADJUST RETURN
6313 035404 000207 RTS PC ;RETURN
6314
6315 035406 005037 001202 63$: CLR $ESCAPE ;CLEAR ESCAPE
6316 035412 012706 001100 MOV #STACK,SP ;FORCE STACK
6317 035416 000177 143466 JMP @SLPERR ;JUMP TO LOOP ADDRESS
6318
6319
6320 .....
6321 .SBTTL SIMULATE ONE BIT OF READ DATA IN MAINTENANCE MODE
6322 035422 105737 003254 RDBIT: TSTB PR.BIT ;CHECK IF ONE
6323 035426 001024 BNE 10$ ;YES, SIMULATE ONE
```

```

6324 035430 105737 003256          TSTB   M1.BIT          ;CHECK IF PREVIOUS ONE
6325 035434 001404          BEQ    4$              ;INSERT TRANSITION
6326 035436 012762 000440 000026  MOV    #DMD!MCLK,RKMR1(R2) ;DO NOT INSERT TRANSITION
6327 035444 000403          BR     5$              ;CLOCK IN ZERO
6328
6329 035446 012762 001440 000026 4$:  MOV    #DMD!MCLK!MERD,RKMR1(R2) ;INSERT TRANSITION
6330 035454 012762 000040 000026 5$:  MOV    #DMD,RKMR1(R2) ;CLOCK IN ZERO
6331 035462 012762 000440 000026  MOV    #DMD!MCLK,RKMR1(R2)
6332 035470 012762 000040 000026  MOV    #DMD,RKMR1(R2)
6333 035476 000207          RTS    PC              ;RETURN
6334
6335 035500 012762 000440 000026 10$: MOV    #DMD!MCLK,RKMR1(R2) ;CLOCK IN ONE
6336 035506 012762 000040 000026  MOV    #DMD,RKMR1(R2)
6337 035514 012762 001440 000026  MOV    #DMD!MCLK!MERD,RKMR1(R2)
6338 035522 012762 000040 000026  MOV    #DMD,RKMR1(R2)
6339 035530 000207          RTS    PC              ;RETURN
6340          .SBTTL  APT COMMUNICATIONS ROUTINE
6341
6342          ;:*****
6343 035532 112737 000001 035776 $ATY1: MOVB   #1,$FFLG          ;;TO REPORT FATAL ERROR
6344 035540 112737 000001 035774 $ATY3: MOVB   #1,$MFLG          ;;TO TYPE A MESSAGE
6345 035546 000403          BR     $ATYC
6346 035550 112737 000001 035776 $ATY4: MOVB   #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
6347 035556          $ATYC:
6348 035556 010046          MOV    R0,-(SP)        ;;PUSH R0 ON STACK
6349 035560 010146          MOV    R1,-(SP)        ;;PUSH R1 ON STACK
6350 035562 105737 035774          TSTB   $MFLG          ;;SHOULD TYPE A MESSAGE?
6351 035566 001450          BEQ    5$              ;;IF NOT: BR
6352 035570 122737 000001 001234  CMPB   #APTENV,$ENV    ;;OPERATING UNDER APT?
6353 035576 001031          BNE   3$              ;;IF NOT: BR
6354 035600 132737 000100 001235  BITB   #APTPOOL,$ENVM  ;;SHOULD SPOOL MESSAGES?
6355 035606 001425          BEQ    3$              ;;IF NOT: BR
6356 035610 017600 000004          MOV    @4(SP),R0       ;;GET MESSAGE ADDR.
6357 035614 062766 000002 000004  ADD    #2,4(SP)        ;;BUMP RETURN ADDR.
6358 035622 005737 001214          1$:  TST    $MSGTYPE       ;;SEE IF DONE W/ LAST XMISSION?
6359 035626 001375          BNE   1$              ;;IF NOT: WAIT
6360 035630 010037 001230          MOV    R0,$MSGAD       ;;PUT ADDR IN MAILBOX
6361 035634 105720          2$:  TSTB   (R0)+          ;;FIND END OF MESSAGE
6362 035636 001376          BNE   2$
6363 035640 163700 001230          SUB    $MSGAD,R0       ;;SUB START OF MESSAGE
6364 035644 006200          ASR    R0              ;;GET MESSAGE LGTH IN WORDS
6365 035646 010037 001232          MOV    R0,$MSGGLT     ;;PUT LENGTH IN MAILBOX
6366 035652 012737 000004 001214  MOV    #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
6367 035660 000413          BR     5$
6368 035662 017637 000004 035706 3$:  MOV    @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
6369 035670 062766 000002 000004  ADD    #2,4(SP)        ;;BUMP RETURN ADDRESS
6370 035676 013746 177776          MOV    177776,-(SP)   ;;PUSH 177776 ON STACK
6371 035702 004737 036200          JSR    PC,$TYPE       ;;CALL TYPE MACRO
6372 035706 000000          4$:  .WORD  0
6373 035710          5$:
6374 035710 105737 035776          10$: TSTB   $FFLG          ;;SHOULD REPORT FATAL ERROR?
6375 035714 001416          BEQ    12$            ;;IF NOT: BR
6376 035716 005737 001234          TST    $ENV           ;;RUNNING UNDER APT?
6377 035722 001413          BEQ    12$            ;;IF NOT: BR
6378 035724 005737 001214          11$: TST    $MSGTYPE       ;;FINISHED LAST MESSAGE?
6379 035730 001375          BNE   11$            ;;IF NOT: WAIT

```

```
6380 035732 017637 000004 001216      MOV    @4(SP), $FATAL    ;;GET ERROR #
6381 035740 062766 000002 000004      ADD    #2,4(SP)         ;;BUMP RETURN ADDR.
6382 035746 005237 001214      INC    $MSGTYPE        ;;TELL APT TO TAKE ERROR
6383 035752 105037 035776      12$:  CLR    $FFLG        ;;CLEAR FATAL FLAG
6384 035756 105037 035775      CLR    $LFLG          ;;CLEAR LOG FLAG
6385 035762 105037 035774      CLR    $MFLG          ;;CLEAR MESSAGE FLAG
6386 035766 012601      MOV    (SP)+,R1        ;;POP STACK INTO R1
6387 035770 012600      MOV    (SP)+,R0        ;;POP STACK INTO R0
6388 035772 000207      RTS    PC              ;;RETURN
6389 035774      000      $MFLG: .BYTE 0        ;;MESSG. FLAG
6390 035775      000      $LFLG: .BYTE 0        ;;LOG FLAG
6391 035776      000      $FFLG: .BYTE 0        ;;FATAL FLAG
6392      036000      .EVEN
6393      000200      APTSIZE=200
6394      000001      APTENV=001
6395      000100      APTSPool=100
6396      000040      APTCSUP=040
6397      .SBTTL  ERROR HANDLER ROUTINE
6398
6399      ;*****
6400      ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
6401      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
6402      ;*AND GO TO TYPERR ON ERROR
6403      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
6404      ;*SW15=1      HALT ON ERROR
6405      ;*SW13=1      INHIBIT ERROR TYPEOUTS
6406      ;*SW10=1      BELL ON ERROR
6407      ;*SW09=1      LOOP ON ERROR
6408      ;*CALL
6409      ;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
6410
6411      036000      $ERROR:
6412      036000 104407      7$:  CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
6413      036002 105237 001103      INCB    $ERFLG        ;;SET THE ERROR FLAG
6414      036006 001775      BEQ     7$            ;;DON'T LET THE FLAG GO TO ZERO
6415      036010 013777 001102 143124      MOV    $STNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
6416      036016 032777 002000 143114      BIT    #BIT10,@SWR    ;;BELL ON ERROR?
6417      036024 001402      BEQ     1$            ;;NO - SKIP
6418      036026 104401 001204      TYPE   , $BELL        ;;RING BELL
6419      036032 005237 001112      1$:  INC    $ERTTL        ;;COUNT THE NUMBER OF ERRORS
6420      036036 011637 001116      MOV    (SP), $ERRPC    ;;GET ADDRESS OF ERROR INSTRUCTION
6421      036042 162737 000002 001116      SUB    #2, $ERRPC
6422      036050 117737 143042 001114      MOV    @ $ERRPC, $ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
6423      036056 032777 020000 143054      BIT    #BIT13,@SWR    ;;SKIP TYPEOUT IF SET
6424      036064 001004      BNE    20$           ;;SKIP TYPEOUTS
6425      036066 004737 033722      JSR    PC, TYPERR     ;;GO TO USER ERROR ROUTINE
6426      036072 104401 001211      TYPE   , $CRLF
6427      036076      20$:
6428      036076 122737 000001 001234      CMP    #APTENV, $ENV   ;;RUNNING IN APT MODE
6429      036104 001007      BNE    2$            ;;NO, SKIP APT ERROR REPORT
6430      036106 113737 001114 036120      MOV    $ITEMB, 21$    ;;SET ITEM NUMBER AS ERROR NUMBER
6431      036114 004737 035550      JSR    PC, $ATY4     ;;REPORT FATAL ERROR TO APT
6432      036120      000      21$: .BYTE 0
6433      036121      000      .BYTE 0
6434      036122 000777      22$: BR    22$        ;;APT ERROR LOOP
6435      036124 005777 143010      2$:  TST    @SWR        ;;HALT ON ERROR
```

```
6436 036130 100002          BPL      3$          ;;SKIP IF CONTINUE
6437 036132 000000          HALT                    ;;HALT ON ERROR!
6438 036134 104407          CKSWR                    ;;TEST FOR CHANGE IN SOFT-SWR
6439 036136 032777 001000 142774 3$: BIT      #BIT09,@SWR    ;;LOOP ON ERROR SWITCH SET?
6440 036144 001402          BEQ      4$          ;;BR IF NO
6441 036146 013716 001110          MOV      $LPERR,(SP)   ;;FUDGE RETURN FOR LOOPING
6442 036152 005737 001202          4$: TST      $ESCAPE    ;;CHECK FOR AN ESCAPE ADDRESS
6443 036156 001402          BEQ      5$          ;;BR IF NONE
6444 036160 013716 001202          MOV      $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
6445 036164                    5$:                    ;;
6446 036164 022737 033150 000042          CMP      #SENDAD,@#42 ;;ACT-11 AUTO-ACCEPT?
6447 036172 001001          BNE      6$          ;;BRANCH IF NO
6448 036174 000000          HALT                    ;;YES
6449 036176                    6$:                    ;;
6450 036176 000002          RTI                      ;;RETURN
```

.SBTTL TYPE ROUTINE

```
6451
6452
6453
6454
6455
6456
6457
6458
6459
6460
6461
6462
6463
6464
6465
6466
6467
6468
6469
6470
6471
6472
6473
6474
6475
6476
6477
6478
6479
6480
6481
6482
6483
6484
6485
6486
6487
6488
6489
6490
6491
```

\*\*\*\*\*  
\*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 \$TFPLG ;;IS THERE A TERMINAL?  
BPL 1\$ ;;BR IF YES  
HALT ;;HALT HERE IF NO TERMINAL  
BR 3\$ ;;LEAVE  
1\$: MOV R0,-(SP) ;;SAVE R0  
MOV @2(SP),R0 ;;GET ADDRESS OF ASCIZ STRING  
CMPB #APTENV,\$ENV ;;RUNNING IN APT MODE  
BNE 62\$ ;;NO,GO CHECK FOR APT CONSOLE  
BITB #APTSPOOL,\$ENVM ;;SPOOL MESSAGE TO APT  
BEQ 62\$ ;;NO,GO CHECK FOR CONSOLE  
MOV R0,61\$ ;;SETUP MESSAGE ADDRESS FOR APT  
JSR PC,\$ATY3 ;;SPOOL MESSAGE TO APT  
61\$: .WORD 0 ;;MESSAGE ADDRESS  
62\$: BITB #APTCSUP,\$ENVM ;;APT CONSOLE SUPPRESSED  
BNE 60\$ ;;YES,SKIP TYPE OUT  
2\$: MOV (R0)+,-(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  
60\$: MOV (SP)+,R0 ;;RESTORE R0  
3\$: ADD #2,(SP) ;;ADJUST RETURN PC  
RTI ;;RETURN  
4\$: CMPB #HT,(SP) ;;BRANCH IF <HT>  
BEQ 8\$

```
6492 036306 122716 000200      CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
6493 036312 001006              BNE     5$              ;;
6494 036314 005726              TST     (SP)+           ;;POP <CR><LF> EQUIV
6495 036316 104401              TYPE                    ;;TYPE A CR AND LF
6496 036320 001211              $CRLF
6497 036322 105037 036456      CLRB    $CHARCNT       ;;CLEAR CHARACTER COUNT
6498 036326 000755              BR      2$              ;;GET NEXT CHARACTER
6499 036330 004737 036412      5$:    JSR    PC,$TYPEC   ;;GO TYPE THIS CHARACTER
6500 036334 123726 001156      6$:    CMPB    $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
6501 036340 001350              BNE     2$              ;;IF NO GO GET NEXT CHAR.
6502 036342 013746 001154      MOV     $NULL,-(SP)    ;;GET # OF FILLER CHARS. NEEDED
6503                                ;;AND THE NULL CHAR.
6504 036346 105366 000001      7$:    DECB    1(SP)     ;;DOES A NULL NEED TO BE TYPED?
6505 036352 002770              BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
6506 036354 004737 036412      JSR    PC,$TYPEC   ;;GO TYPE A NULL
6507 036360 105337 036456      DECB    $CHARCNT      ;;DO NOT COUNT AS A COUNT
6508 036364 000770              BR      7$              ;;LOOP
```

```
6509                                ;HORIZONTAL TAB PROCESSOR
6510
6511
6512 036366 112716 000040      8$:    MOVB    #' ,(SP)  ;;REPLACE TAB WITH SPACE
6513 036372 004737 036412      9$:    JSR    PC,$TYPEC   ;;TYPE A SPACE
6514 036376 132737 000007 036456  BITB    #7,$CHARCNT    ;;BRANCH IF NOT AT
6515 036404 001372              BNE     9$              ;;TAB STOP
6516 036406 005726              TST     (SP)+           ;;POP SPACE OFF STACK
6517 036410 000724              BR      2$              ;;GET NEXT CHARACTER
6518 036412 105777 142532      $TYPEC: TSTB    @STPS     ;;WAIT UNTIL PRINTER IS READY
6519 036416 100375              BPL     $TYPEC
6520 036420 116677 000002 142524  MOVB    2(SP),@STPB    ;;LOAD CHAR TO BE TYPED INTO DATA REG.
6521 036426 122766 000015 000002  CMPB    #CR,2(SP)     ;;IS CHARACTER A CARRIAGE RETURN?
6522 036434 001003              BNE     1$              ;;BRANCH IF NO
6523 036436 105037 036456      CLRB    $CHARCNT       ;;YES--CLEAR CHARACTER COUNT
6524 036442 000406              BR      $TYPEX         ;;EXIT
6525 036444 122766 000012 000002  1$:    CMPB    #LF,2(SP)  ;;IS CHARACTER A LINE FEED?
6526 036452 001402              BEQ     $TYPEX         ;;BRANCH IF YES
6527 036454 105227              INCB    (PC)+          ;;COUNT THE CHARACTER
6528 036456 000000      $CHARCNT: .WORD 0      ;;CHARACTER COUNT STORAGE
6529 036460 000207      $TYPEX: RTS    PC
```

```
6530
6531                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
6532
6533                                ;*****
6534                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
6535                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
6536                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
6537                                ;*CALL:
6538                                ;*      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
6539                                ;*      TYPOS                    ;;CALL FOR TYPEOUT
6540                                ;*      .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
6541                                ;*      .BYTE  M              ;;M=1 OR 0
6542                                ;*                                ;;1=TYPE LEADING ZEROS
6543                                ;*                                ;;0=SUPPRESS LEADING ZEROS
6544
6545                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
6546                                ;*$TYPOS OR $TYPOC
6547                                ;*CALL:
```



```

6548      *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
6549      *      TYPON      ;;CALL FOR TYPEOUT
6550      *
6551      *      *STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
6552      *      *CALL:
6553      *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
6554      *      TYPOC      ;;CALL FOR TYPEOUT
6555
6556 036462 017646 000000      $TYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
6557 036466 116637 000001      036705  MOVVB    1(SP),$OFILL      ;;LOAD ZERO FILL SWITCH
6558 036474 112637 036707      MOVVB    (SP)+,$OMODE+1      ;;NUMBER OF DIGITS TO TYPE
6559 036500 062716 000002      ADD      #2,(SP)      ;;ADJUST RETURN ADDRESS
6560 036504 000406      BR      $TYPON
6561 036506 112737 000001      036705 $TYPOC: MOVVB    #1,$OFILL      ;;SET THE ZERO FILL SWITCH
6562 036514 112737 000006      036707  MOVVB    #6,$OMODE+1      ;;SET FOR SIX(6) DIGITS
6563 036522 112737 000005      036704 $TYPON: MOVVB    #5,$OCNT      ;;SET THE ITERATION COUNT
6564 036530 010346      MOV      R3,-(SP)      ;;SAVE R3
6565 036532 010446      MOV      R4,-(SP)      ;;SAVE R4
6566 036534 010546      MOV      R5,-(SP)      ;;SAVE R5
6567 036536 113704 036707      MOVVB    $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
6568 036542 005404      NEG      R4
6569 036544 062704 000006      ADD      #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
6570 036550 110437 036706      MOVVB    R4,$OMODE      ;;SAVE IT FOR USE
6571 036554 113704 036705      MOVVB    $OFILL,R4      ;;GET THE ZERO FILL SWITCH
6572 036560 016605 000012      MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
6573 036564 005003      CLR      R3      ;;CLEAR THE OUTPUT WORD
6574 036566 006105      1$:    ROL      R5      ;;RO'ATE MSB INTO 'C'
6575 036570 000404      BR      3$      ;;GO DO MSB
6576 036572 006105      2$:    ROL      R5      ;;FORM THIS DIGIT
6577 036574 006105      ROL      R5
6578 036576 006105      ROL      R5
6579 036600 010503      MOV      R5,R3
6580 036602 006103      3$:    ROL      R3      ;;GET LSB OF THIS DIGIT
6581 036604 105337 036706      DECB    $OMODE      ;;TYPE THIS DIGIT?
6582 036610 100016      BPL      7$      ;;BR IF NO
6583 036612 042703 177770      BIC      #177770,R3      ;;GET RID OF JUNK
6584 036616 001002      BNE      4$      ;;TEST FOR 0
6585 036620 005704      TST      R4      ;;SUPPRESS THIS 0?
6586 036622 001403      BEQ      5$      ;;BR IF YES
6587 036624 005204      4$:    INC      R4      ;;DON'T SUPPRESS ANYMORE 0'S
6588 036626 052703 000060      BIS      #'0,R3      ;;MAKE THIS DIGIT ASCII
6589 036632 052703 000040      5$:    BIS      #' ,R3      ;;MAKE ASCII IF NOT ALREADY
6590 036636 110337 036702      MOVVB    R3,8$      ;;SAVE FOR TYPING
6591 036642 104401 036702      TYPE    ,8$      ;;GO TYPE THIS DIGIT
6592 036646 105337 036704      7$:    DECB    $OCNT      ;;COUNT BY 1
6593 036652 003347      BGT      2$      ;;BR IF MORE TO DO
6594 036654 002402      BLT      6$      ;;BR IF DONE
6595 036656 005204      INC      R4      ;;INSURE LAST DIGIT ISN'T A BLANK
6596 036660 000744      BR      2$      ;;GO DO THE LAST DIGIT
6597 036662 012605      6$:    MOV      (SP)+,R5      ;;RESTORE R5
6598 036664 012604      MOV      (SP)+,R4      ;;RESTORE R4
6599 036666 012603      MOV      (SP)+,R3      ;;RESTORE R3
6600 036670 016666 000002 000004      MOV      2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
6601 036676 012616      MOV      (SP)+,(SP)
6602 036700 000002      RTI
6603 036702 000      8$:    .BYTE 0      ;;RETURN
        ;;STORAGE FOR ASCII DIGIT

```

6604 036703 000  
6605 036704 000  
6606 036705 000  
6607 036706 000000  
6608  
6609  
6610  
6611  
6612  
6613  
6614  
6615  
6616  
6617  
6618  
6619  
6620 036710  
6621 036710 010046  
6622 036712 010146  
6623 036714 010246  
6624 036716 010346  
6625 036720 010546  
6626 036722 012746 020200  
6627 036726 016605 000020  
6628 036732 100004  
6629 036734 005405  
6630 036736 112766 000055 000001  
6631 036744 005000 1\$:  
6632 036746 012703 037124  
6633 036752 112723 000040  
6634 036756 005002 2\$:  
6635 036760 016001 037114  
6636 036764 160105 3\$:  
6637 036766 002402  
6638 036770 005202  
6639 036772 000774  
6640 036774 060105 4\$:  
6641 036776 005702  
6642 037000 001002  
6643 037002 105716  
6644 037004 100407  
6645 037006 106316 5\$:  
6646 037010 103003  
6647 037012 116663 000001 177777  
6648 037020 052702 000060 6\$:  
6649 037024 052702 000040 7\$:  
6650 037030 110223  
6651 037032 005720  
6652 037034 020027 000010  
6653 037040 002746  
6654 037042 003002  
6655 037044 010502  
6656 037046 000764  
6657 037050 105726 8\$:  
6658 037052 100003  
6659 037054 116663 177777 177776

```

      .BYTE 0          ;; TERMINATOR FOR TYPE ROUTINE
$OCNT: .BYTE 0        ;; OCTAL DIGIT COUNTER
$OFILL: .BYTE 0       ;; ZERO FILL SWITCH
$OMODE: .WORD 0       ;; NUMBER OF DIGITS TO TYPE
.SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
*REPLACED WITH SPACES.
*CALL:
*      MOV     NUM,-(SP)      ;; PUT THE BINARY NUMBER ON THE STACK
*      TYPDS                    ;; GO TO THE ROUTINE

$TYPDS:
MOV     R0,-(SP)           ;; PUSH R0 ON STACK
MOV     R1,-(SP)           ;; PUSH R1 ON STACK
MOV     R2,-(SP)           ;; PUSH R2 ON STACK
MOV     R3,-(SP)           ;; PUSH R3 ON STACK
MOV     R5,-(SP)           ;; PUSH R5 ON STACK
MOV     #20200,-(SP)       ;; SET BLANK SWITCH AND SIGN
MOV     20(SP),R5          ;; GET THE INPUT NUMBER
BPL     1$                 ;; BR IF INPUT IS POS.
NEG     R5                 ;; MAKE THE BINARY NUMBER POS.
MOVB   #'-,1(SP)          ;; MAKE THE ASCII NUMBER NEG.
1$:    CLR     R0           ;; ZERO THE CONSTANTS INDEX
MOV     #$DBLK,R3         ;; SETUP THE OUTPUT POINTER
MOVB   #' ,(R3)+          ;; SET THE FIRST CHARACTER TO A BLANK
2$:    CLR     R2           ;; CLEAR THE BCD NUMBER
MOV     $DTBL(R0),R1      ;; GET THE CONSTANT
3$:    SUB     R1,R5        ;; FORM THIS BCD DIGIT
BLT     4$                 ;; BR IF DONE
INC     R2                 ;; INCREASE THE BCD DIGIT BY 1
BR      3$
4$:    ADD     R1,R5        ;; ADD BACK THE CONSTANT
TST     R2                 ;; CHECK IF BCD DIGIT=0
BNE     5$                 ;; FALL THROUGH IF 0
TSTB   (SP)               ;; STILL DOING LEADING 0'S?
BMI     7$                 ;; BR IF YES
5$:    ASLB   (SP)         ;; MSD?
BCC     6$                 ;; BR IF NO
MOVB   1(SP),-1(R3)       ;; YES--SET THE SIGN
6$:    BIS   #'0,R2        ;; MAKE THE BCD DIGIT ASCII
7$:    BIS   #' ,R2        ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB   R2,(R3)+          ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST    (R0)+              ;; JUST INCREMENTING
CMP     R0,#10            ;; CHECK THE TABLE INDEX
BLT     2$                 ;; GO DO THE NEXT DIGIT
BGT     8$                 ;; GO TO EXIT
MOV     R5,R2             ;; GET THE LSD
BR      6$                 ;; GO CHANGE TO ASCII
8$:    TSTB  (SP)+         ;; WAS THE LSD THE FIRST NON-ZERO?
BPL     9$                 ;; BR IF NO
9$:    MOVB  -1(SP),-2(R3) ;; YES--SET THE SIGN FOR TYPING

```

```
6660 037062 105013          9$:   CLRB   (R3)           ;;SET THE TERMINATOR
6661 037064 012605          MOV   (SP)+,R5         ;;POP STACK INTO R5
6662 037066 012603          MOV   (SP)+,R3         ;;POP STACK INTO R3
6663 037070 012602          MOV   (SP)+,R2         ;;POP STACK INTO R2
6664 037072 012601          MOV   (SP)+,R1         ;;POP STACK INTO R1
6665 037074 012600          MOV   (SP)+,R0         ;;POP STACK INTO R0
6666 037076 104401 037124    TYPE   $DBLK           ;;NOW TYPE THE NUMBER
6667 037102 016666 000002 000004  MOV   2(SP),4(SP)     ;;ADJUST THE STACK
6668 037110 012616          MOV   (SP)+,(SP)
6669 037112 000002          RTI                    ;;RETURN TO USER
6670 037114 023420          $DTBL: 10000.
6671 037116 001750          1000.
6672 037120 000144          100.
6673 037122 000012          10.
6674 037124 000004          $DBLK: .BLKW 4
6675                                     .SBTTL TTY INPUT ROUTINE
6676
6677                                     ;:*****
6678                                     .ENABL LSB
6679 037134 000000          $TKCNT: .WORD 0       ;;NUMBER OF ITEMS IN QUEUE
6680 037136 000000          $TKQIN: .WORD 0       ;;INPUT POINTER
6681 037140 000000          $TKQOUT: .WORD 0      ;;OUTPUT POINTER
6682 037142 000001          $TKQSRV: .BLKB 1     ;;TTY KEYBOARD QUEUE
6683                                     $TKQEND=.
6684                                     .EVEN
6685
6686                                     ;*TK INITIALIZE ROUTINE
6687                                     ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
6688                                     ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
6689
6690                                     ;*CALL:
6691                                     ;*   JSR   PC,$TKINT
6692                                     ;*   RETURN
6693
6694 037144 005037 037134          $TKINT: CLR   $TKCNT     ;;CLEAR COUNT OF ITEMS IN QUEUE
6695 037150 012737 037142 037136  MOV   #$TKQSRV,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
6696 037156 013737 037136 037140  MOV   $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
6697 037164 012737 037214 000060  MOV   #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
6698 037172 012737 000200 000062  MOV   #200,@#TKVEC+2  ;;'BR' LEVEL 4
6699 037200 005777 141742          TST   @#TKB           ;;CLEAR DONE FLAG
6700 037204 012777 000100 141732  MOV   #100,@#TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
6701 037212 000207          RTS   PC              ;;RETURN TO CALLER
6702
6703                                     ;*TK SERVICE ROUTINE
6704                                     ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
6705                                     ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
6706                                     ;*IT IN THE QUEUE.
6707                                     ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
6708                                     ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)
6709
6710 037214 117746 141726          $TKSRV: MOVB  @#TKB,-(SP) ;;PICKUP THE CHARACTER
6711 037220 042716 177600          BIC   #^C177,(SP)    ;;STRIP THE JUNK
6712 037224 021627 000003          CMP   (SP),#3        ;;IS IT A CONTROL C?
6713 037230 001007          BNE   1$             ;;BRANCH IF NO
6714 037232 104401 040330          TYPE  ,%CNTLC        ;;TYPE A CONTROL-C (^C)
6715 037236 004737 037144          JSR   PC,$TKINT     ;;INIT THE KEYBOARD
```

```
6716 037242 005726          TST      (SP)+          ;;CLEAN UP STACK
6717 037244 000137 033170    JMP      CTRHLT        ;;CONTROL C RESTART
6718 037250 021627 000007    1$:    CMP      (SP),#7     ;;IS IT A CONTROL G?
6719 037254 001004          BNE      2$           ;;BRANCH IF NO
6720 037256 022737 000176 001140    CMP      #SWREG,SWR    ;;IS SOFT-SWR SELECTED?
6721 037264 001500          BEQ      6$           ;;GO TO SWR CHANGE
6722
6723 037266          2$:
6724 037266 022737 000001 037134    CMP      #1,$TKCNT    ;;IS THE QUEUE FULL?
6725 037274 001004          BNE      3$           ;;BRANCH IF NO
6726 037276 104401 001204    TYPE     ,$BELL       ;;RING THE TTY BELL
6727 037302 005726          TST      (SP)+          ;;CLEAN CHARACTER OFF OF STACK
6728 037304 000451          BR       5$           ;;EXIT
6729 037306 021627 000023    3$:    CMP      (SP),#23     ;;IS IT A CONTROL-S?
6730 037312 001021          BNE      32$          ;;BRANCH IF NO
6731 037314 005077 141624    CLR      @TKS         ;;DISABLE TTY KEYBOARD INTERRUPTS
6732 037320 005726          TST      (SP)+          ;;CLEAN CHAR OFF STACK
6733 037322 105777 141616    31$:    TSTB    @TKS         ;;WAIT FOR A CHAR
6734 037326 100375          BPL      31$         ;;LOOP UNTIL ITS THERE
6735 037330 117746 141612    MOVB    @TKB,-(SP)    ;;GET THE CHARACTER
6736 037334 042716 177600    BIC     #^C177,(SP)  ;;MAKE IT 7-BIT ASCII
6737 037340 022627 000021    CMP      (SP)+,#21    ;;IS IT A CONTROL-Q?
6738 037344 001366          BNE      31$         ;;BRANCH IF NO
6739 037346 012777 000100 141570    MOV      #100,@TKS    ;;REENABLE TTY KEYBOARD INTERRUPTS
6740 037354 000002          RTI
6741 037356 005237 037134    32$:    INC      $TKCNT      ;;COUNT THIS CHARACTER
6742 037362 021627 000140    CMP      (SP),#140   ;;IS IT UPPER CASE?
6743 037366 002405          BLT      4$           ;;BRANCH IF YES
6744 037370 021627 000175    CMP      (SP),#175   ;;IS IT A SPECIAL CHAR?
6745 037374 003002          BGT      4$           ;;BRANCH IF YES
6746 037376 042716 000040    BIC     #40,(SP)     ;;MAKE IT UPPER CASE
6747 037402 112677 177530    4$:    MOVB    (SP)+,@TKQIN  ;;AND PUT IT IN QUEUE
6748 037406 005237 037136    INC      $TKQIN      ;;UPDATE THE POINTER
6749 037412 023727 037136 037143    CMP      $TKQIN,$TKQEND ;;GO OFF THE END?
6750 037420 001003          BNE      5$           ;;BRANCH IF NO
6751 037422 012737 037142 037136    MOV      #$TKQSRRT,$TKQIN ;;RESET THE POINTER
6752 037430 000002          5$:    RTI
6753
6754          ;;*****
6755          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
6756          ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
6757          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
6758          ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
6759 037432 022737 000176 001140    $CKSWR: CMP      #SWREG,SWR    ;;IS THE SOFT-SWR SELECTED
6760 037440 001124          BNE      15$         ;;EXIT IF NOT
6761 037442 105777 141476    TSTB    @TKS         ;;IS A CHAR WAITING?
6762 037446 100121          BPL      15$         ;;IF NOT, EXIT
6763 037450 117746 141472    MOVB    @TKB,-(SP)    ;;YES
6764 037454 042716 177600    BIC     #^C177,(SP)  ;;MAKE IT 7-BIT ASCII
6765 037460 021627 000007    CMP      (SP),#7     ;;IS IT A CONTROL-G?
6766 037464 001300          BNE      2$           ;;IF NOT, PUT IT IN THE TTY QUEUE
6767          ;;AND EXIT
6768
6769          ;;*****
6770          ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
6771          ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
```

```

6772 ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
6773 037466 123727 001134 000001 6$:  CMPB  $AUTOB,#1  ;;ARE WE RUNNING IN AUTO-MODE?
6774 037474 001674  BEQ  2$  ;;BRANCH IF YES
6775 037476 005726  TST  (SP)+  ;;CLEAR CONTROL-G OFF STACK
6776 037500 004737 037144  JSR  PC,$TKINT  ;;FLUSH THE TTY INPUT QUEUE
6777 037504 005077 141434  CLR  @STKS  ;;DISABLE TTY KEYBOARD INTERRUPTS
6778 037510 112737 000001 001135  MOVB #1,$INTAG  ;;SET INTERRUPT MODE INDICATOR
6779
6780 037516 104401 040342  TYPE  ,$CNTLG  ;;ECHO THE CONTROL-G (^G)
6781 037522 104401 040347  SGTSWR: TYPE  ,$MSWR  ;;TYPE CURRENT CONTENTS
6782 037526 013746 000176  MOV  SWREG,-(SP)  ;;SAVE SWREG FOR TYPEOUT
6783 037532 104402  TYPOC  ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
6784 037534 104401 040360  TYPE  ,$MNEW  ;;PROMPT FOR NEW SWR
6785 037540 005046 19$:  CLR  -(SP)  ;;CLEAR COUNTER
6786 037542 005046  CLR  -(SP)  ;;THE NEW SWR
6787 037544 105777 141374  7$:  TSTB @STKS  ;;CHAR THERE?
6788 037550 100375  BPL  7$  ;;IF NOT TRY AGAIN
6789
6790 037552 117746 141370  MOVB @STKB,-(SP)  ;;PICK UP CHAR
6791 037556 042716 177600  BIC  #^C177,(SP)  ;;MAKE IT 7-BIT ASCII
6792
6793 037562 021627 000003  CMP  (SP),#3  ;;IS IT A CONTROL-C?
6794 037566 001015  BNE  9$  ;;BRANCH IF NOT
6795 037570 104401 040330  TYPE  ,$CNTLC  ;;YES, ECHO CONTROL-C (^C)
6796 037574 062706 000006  ADD  #6,SP  ;;CLEAN UP STACK
6797 037600 123727 001135 000001  CMPB $INTAG,#1  ;;REENABLE TTY KEYBOARD INTERRUPTS?
6798 037606 001003  BNE  8$  ;;BRANCH IF NO
6799 037610 012777 000100 141326  MOV  #100,@STKS  ;;ALLOW TTY KEYBOARD INTERRUPTS
6800 037616 000137 033170  8$:  JMP  CTRHLT  ;;CONTROL-C RESTART
6801
6802
6803 037622 021627 000025  9$:  CMP  (SP),#25  ;;IS IT A CONTROL-U?
6804 037626 001005  BNE  10$  ;;BRANCH IF NOT
6805 037630 104401 040335  TYPE  ,$CNTLU  ;;YES, ECHO CONTROL-U (^U)
6806 037634 062706 000006  20$:  ADD  #6,SP  ;;IGNORE PREVIOUS INPUT
6807 037640 000737  BR  19$  ;;LET'S TRY IT AGAIN
6808
6809
6810 037642 021627 000015  10$:  CMP  (SP),#15  ;;IS IT A <CR>?
6811 037646 001022  BNE  16$  ;;BRANCH IF NO
6812 037650 005766 000004  TST  4(SP)  ;;YES, IS IT THE FIRST CHAR?
6813 037654 001403  BEQ  11$  ;;BRANCH IF YES
6814 037656 016677 000002 141254  MOV  2(SP),@SWR  ;;SAVE NEW SWR
6815 037664 062706 000006  11$:  ADD  #6,SP  ;;CLEAR UP STACK
6816 037670 104401 001211  14$:  TYPE  ,$CRLF  ;;ECHO <CR> AND <LF>
6817 037674 123727 001135 000001  CMPB $INTAG,#1  ;;RE-ENABLE TTY KBD INTERRUPTS?
6818 037702 001003  BNE  15$  ;;BRANCH IF NOT
6819 037704 012777 000100 141232  MOV  #100,@STKS  ;;RE-ENABLE TTY KBD INTERRUPTS
6820 037712 000002  15$:  RTI  ;;RETURN
6821 037714 004737 036412  16$:  JSR  PC,$TYPEC  ;;ECHO CHAR
6822 037720 021627 000060  CMP  (SP),#60  ;;CHAR < 0?
6823 037724 002420  BLT  18$  ;;BRANCH IF YES
6824 037726 021627 000067  CMP  (SP),#67  ;;CHAR > 7?
6825 037732 003015  BGT  18$  ;;BRANCH IF YES
6826 037734 042726 000060  BIC  #60,(SP)+  ;;STRIP-OFF ASCII
6827 037740 005766 000002  TST  2(SP)  ;;IS THIS THE FIRST CHAR

```

```
6828 037744 001403          BEQ      17$          ;;BRANCH IF YES
6829 037746 006316          ASL      (SP)        ;;NO, SHIFT PRESENT
6830 037750 006316          ASL      (SP)        ;;CHAR OVER TO MAKE
6831 037752 006316          ASL      (SP)        ;;ROOM FOR NEW ONE.
6832 037754 005266 000002    17$: INC      2(SP)     ;;KEEP COUNT OF CHAR
6833 037760 056616 177776    BIS      -2(SP),(SP) ;;SET IN NEW CHAR
6834 037764 000667          BR       7$          ;;GET THE NEXT ONE
6835 037766 104401 001210    18$: TYPE   $QUES     ;;TYPE ?<CR><LF>
6836 037772 000720          BR       20$        ;;SIMULATE CONTROL-U
6837
6838
6839
6840
6841
6842
6843
6844
6845
6846
6847
6848 037774 011646          $RDCHR: MOV      (SP),-(SP) ;;PUSH DOWN THE PC AND
6849 037776 016666 000004 000002  MOV      4(SP),2(SP) ;;THE PS
6850 040004 005066 000004          CLR      4(SP)     ;;GET READY FOR A CHARACTER
6851 040010 005046          CLR      -(SP)    ;;PUT NEW PS ON STACK
6852 040012 012746 040020          MOV      #64$,-(SP) ;;PUT NEW PC ON STACK
6853 040016 000002          RTI          ;;POP NEW PC AND PS
6854 040020
6855 040020 005737 037134    64$: TST      $TKCNT   ;;WAIT ON A CHARACTER
6856 040024 001775          1$: BEQ      1$          ;;
6857 040026 005337 037134          DEC      $TKCNT   ;;DECREMENT THE COUNTER
6858 040032 117766 177102 000004  MOVB     @($TKQOUT),4(SP) ;;GET ONE CHARACTER
6859 040040 005237 037140          INC      $TKQOUT  ;;UPDATE THE POINTER
6860 040044 023727 037140 037143  CMP      $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
6861 040052 001003          BNE      2$          ;;BRANCH IF NO
6862 040054 012737 037142 037140  MOV      #$TKQSRT,$TKQOUT ;;RESET THE POINTER
6863 040062 000002          2$: RTI          ;;RETURN
6864
6865
6866
6867
6868
6869
6870
6871 040064 010346          $RDLIN: MOV      R3,-(SP) ;;SAVE R3
6872 040066 005046          CLR      -(SP)    ;;CLEAR THE RUBOUT KEY
6873 040070 012703 040320    1$: MOV      #$TTYIN,R3 ;;GET ADDRESS
6874 040074 022703 040330    2$: CMP      #$TTYIN+8.,R3 ;;BUFFER FULL?
6875 040100 101456          BLOS     4$          ;;BR IF YES
6876 040102 104410          RDCHR    ;;GO READ ONE CHARACTER FROM THE TTY
6877 040104 112613          MOVB     (SP)+,(R3) ;;GET CHARACTER
6878 040106 122713 000177    10$: CMPB     #177,(R3) ;;IS IT A RUBOUT
6879 040112 001022          BNE      5$          ;;BR IF NO
6880 040114 005716          TST      (SP)     ;;IS THIS THE FIRST RUBOUT?
6881 040116 001007          BNE      6$          ;;BR IF NO
6882 040120 112737 000134 040316  MOVB     #'\,9$    ;;TYPE A BACK SLASH
6883 040126 104401 040316          TYPE     ,9$
```

```

6884 040132 012716 177777
6885 040136 005303
6886 040140 020327 040320
6887 040144 103434
6888 040146 111337 040316
6889 040152 104401 040316
6890 040156 000746
6891 040160 005716
6892 040162 001406
6893 040164 112737 000134 040316
6894 040172 104401 040316
6895 040176 005016
6896 040200 122713 000025
6897 040204 001003
6898 040206 104401 040335
6899 040212 000726
6900 040214 122713 000022
6901 040220 001011
6902 040222 105013
6903 040224 104401 001211
6904 040230 104401 040320
6905 040234 000717
6906 040236 104401 001210
6907 040242 000712
6908 040244 111337 040316
6909 040250 104401 040316
6910 040254 122723 000015
6911 040260 001305
6912 040262 105063 177777
6913 040266 104401 001212
6914 040272 005726
6915 040274 012603
6916 040276 011646
6917 040300 016666 000004 000002
6918 040306 012766 040320 000004
6919 040314 000002
6920 040316 000
6921 040317 000
6922 040320 000010
6923 040330 041536 005015 000
6924 040335 136 006525 000012
6925 040342 043536 005015 000
6926 040347 015 051412 051127
6927 040354 036440 000040
6928 040360 020040 042516 020127
6929 040366 020075 000
6930 040372
6931
6932
6933
6934
6935
6936
6937
6938
6939

MOV #-1,(SP) ::SET THE RUBOUT KEY
6$: DEC R3 ::BACKUP BY ONE
CMP R3,#$TTYIN ::STACK EMPTY?
BLO 4$ ::BR IF YES
MOV (R3),9$ ::SETUP TO TYPEOUT THE DELETED CHAR.
TYPE ,9$ ::GO TYPE
BR 2$ ::GO READ ANOTHER CHAR.
5$: TST (SP) ::RUBOUT KEY SET?
BEQ 7$ ::BR IF NO
MOV #'\,9$ ::TYPE A BACK SLASH
TYPE ,9$
CLR (SP) ::CLEAR THE RUBOUT KEY
7$: CMPB #25,(R3) ::IS CHARACTER A CTRL U?
BNE 8$ ::BR IF NO
TYPE ,SCNTLU ::TYPE A CONTROL 'U'
BR 1$ ::GO START OVER
8$: CMPB #22,(R3) ::IS CHARACTER A '^R'?
BNE 3$ ::BRANCH IF NO
CLRB (R3) ::CLEAR THE CHARACTER
TYPE ,SCRLF ::TYPE A 'CR' & 'LF'
TYPE ,TTYIN ::TYPE THE INPUT STRING
BR 2$ ::GO PICKUP ANOTHER CHACTER
4$: TYPE ,SQUES ::TYPE A '?'
BR 1$ ::CLEAR THE BUFFER AND LOOP
3$: MOV (R3),9$ ::ECHO THE CHARACTER
TYPE ,9$
CMPB #15,(R3)+ ::CHECK FOR RETURN
BNE 2$ ::LOOP IF NOT RETURN
CLRB -1(R3) ::CLEAR RETURN (THE 15)
TYPE ,SLF ::TYPE A LINE FEED
TST (SP)+ ::CLEAN RUBOUT KEY FROM THE STACK
MOV (SP)+,R3 ::RESTORE R3
MOV (SP),-(SP) ::ADJUST THE STACK AND PUT ADDRESS OF THE
MOV 4(SP),2(SP) :: FIRST ASCII CHARACTER ON IT
MOV #TTYIN,4(SP)
RTI ::RETURN
9$: .BYTE 0 ::STORAGE FOR ASCII CHAR. TO TYPE
.BYTE 0 ::TERMINATOR
$TTYIN: .BLKB 8. ::RESERVE 8 BYTES FOR TTY INPUT
$CNTLC: .ASCIZ /^C/<15><12> ::CONTROL 'C'
$CNTLU: .ASCIZ /^U/<15><12> ::CONTROL 'U'
$CNTLG: .ASCIZ /^G/<15><12> ::CONTROL 'G'
$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.

```

```
6940      ;*CALL:
6941      ;*      RDOCT          ;;READ AN OCTAL NUMBER
6942      ;*      RETURN HERE    ;;LOW ORDER BITS ARE CN TOP OF THE STACK
6943      ;*                               ;;HIGH ORDER BITS ARE IN $HIOCT
6944
6945 040372 011646      $RDOCT: MOV      (SP),-(SP)    ;;PROVIDE SPACE FOR THE
6946 040374 016666 000004 000002  MOV      4(SP),2(SP)    ;;INPUT NUMBER
6947 040402 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
6948 040404 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
6949 040406 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
6950 040410 104411      1$:      RDLIN          ;;READ AN ASCIZ LINE
6951 040412 012600      MOV      (SP)+,R0      ;;GET ADDRESS OF 1ST CHARACTER
6952 040414 010037 040520  MOV      R0,5$         ;;AND SAVE IT
6953 040420 005001      CLR      R1            ;;CLEAR DATA WORD
6954 040422 005002      CLR      R2
6955 040424 112046      2$:      MOVB      (R0)+,-(SP)  ;;PICKUP THIS CHARACTER
6956 040426 001420      BEQ      3$           ;;IF ZERO GET OUT
6957 040430 122716 000060  CMPB     #'0,(SP)      ;;MAKE SURE THIS CHARACTER
6958 040434 003026      BGT      4$           ;;IS AN OCTAL DIGIT
6959 040436 122716 000067  CMPB     #'7,(SP)
6960 040442 002423      BLT      4$
6961 040444 006301      ASL      R1            ;;*2
6962 040446 006102      ROL      R2
6963 040450 006301      ASL      R1            ;;*4
6964 040452 006102      ROL      R2
6965 040454 006301      ASL      R1            ;;*8
6966 040456 006102      ROL      R2
6967 040460 042716 177770  BIC      #'C7,(SP)     ;;STRIP THE ASCII JUNK
6968 040464 062601      ADD      (SP)+,R1      ;;ADD IN THIS DIGIT
6969 040466 000756      BR       2$           ;;LOOP
6970 040470 005726      3$:      TST      (SP)+       ;;CLEAN TERMINATOR FROM STACK
6971 040472 010166 000012  MOV      R1,12(SP)     ;;SAVE THE RESULT
6972 040476 010237 040530  MOV      R2,$HIOCT
6973 040502 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
6974 040504 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
6975 040506 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
6976 040510 000002      RTI
6977 040512 005726      4$:      TST      (SP)+       ;;CLEAN PARTIAL FROM STACK
6978 040514 105010      CLRB     (R0)         ;;SET A TERMINATOR
6979 040516 104401      TYPE
6980 040520 000000      5$:      .WORD     0           ;;TYPE UP THRU THE BAD CHAR.
6981 040522 104401 001210  TYPE     , $QUES      ;;'"' 'CR' & 'LF'
6982 040526 000730      BR       1$           ;;TRY AGAIN
6983 040530 000000      $HIOCT: .WORD     0           ;;HIGH ORDER BITS GO HERE
6984      .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
6985
6986      ;*****
6987      ;*SAVE R0-R5
6988      ;*CALL:
6989      ;*      SAVREG
6990      ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
6991      ;*
6992      ;*TOP---(+16)
6993      ;* +2---(+18)
6994      ;* +4---R5
6995      ;* +6---R4
```



```
6996      ;* +8---R3
6997      ;*+10---R2
6998      ;*+12---R1
6999      ;*+14---R0
7000
7001 040532      $SAVREG:
7002 040532 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
7003 040534 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
7004 040536 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
7005 040540 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
7006 040542 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
7007 040544 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
7008 040546 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PS OF MAIN FLOW
7009 040552 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PC OF MAIN FLOW
7010 040556 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PS OF CALL
7011 040562 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PC OF CALL
7012 040566 000002      RTI
7013
7014      ;*RESTORE R0-R5
7015      ;*CALL:
7016      ;*      RESREG
7017      $RESREG:
7018 040570 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PC OF CALL
7019 040574 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PS OF CALL
7020 040600 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PC OF MAIN FLOW
7021 040604 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PS OF MAIN FLOW
7022 040610 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
7023 040612 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
7024 040614 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
7025 040616 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
7026 040620 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
7027 040622 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
7028 040624 000002      RTI
7029
7030      .SBTTL POWER DOWN AND UP ROUTINE
7031
7032      ;;*****
7033
7034      ;POWER DOWN ROUTINE
7035 040626 017737 140306 003312 $PWRDN: MOV      @SWR,SAVSWR      ;SAVE SWITCH REGISTER
7036 040634 012737 040654 000024      MOV      #SPWRUP,PWRVEC      ;SET UP VECTOR
7037 040642 012737 000340 000026      MOV      #PR7,PWRVEC+2
7038 040650 000000      HALT
7039 040652 000776      BR      -2      ;HANG UP
7040
7041      ;;*****
7042
7043      ;POWER UP ROUTINE
7044 040654 005037 040744      $PWRUP: CLR      $PWRCT      ;LOAD WAIT COUNT
7045 040660 012737 000144 040746      MOV      #100,$PWRCT+2
7046 040666 005237 040744      1$: INC      $PWRCT      ;WAIT FOR TELETYPE
7047 040672 001375      BNE      1$
7048 040674 005337 040746      DEC      $PWRCT+2
7049 040700 001372      BNE      1$
7050 040702 012737 040626 000024      MOV      #SPWRDN,PWRVEC      ;SET UP FOR POWER DOWN VECTOR
7051 040710 012737 000340 000026      MOV      #PR7,PWRVEC+2
```

```

7052 040716 012706 001100      MOV    #STACK,SP      ;FORCE STACK
7053 040722 104401 040750      TYPE   $POWER        ;TYPE POWER
7054 040726 013777 003312 140204  MOV    $AVSWR,@SWR    ;RESTORE SWITCH REGISTER
7055 040734 013702 001270      MOV    $BASE,R2      ;REINITIALISE R2 FOR '611 BASE
7056 040740 000177 140142      JMP    @SLPADR ;GO BACK TO LAST TEST
7057
7058 040744 000000 000000      $PWRCT: .WORD 0,0    ;TELETYPE TIME OUT
7059 040750 047520 042527 000122 $POWER: .ASCIZ /POWER/
7060      .EVEN
7061      .SBTTL TRAP DECODER
7062
7063      ;*****
7064      ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
7065      ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
7066      ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
7067      ;*GO TO THAT ROUTINE.
7068
7069 040756 010046      $TRAP: MOV    R0,-(SP)    ;;SAVE R0
7070 040760 016600 000002      MOV    2(SP),R0      ;;GET TRAP ADDRESS
7071 040764 005740      TST    -(R0)         ;;BACKUP BY 2
7072 040766 111000      MOVB   (R0),R0       ;;GET RIGHT BYTE OF TRAP
7073 040770 006300      ASL    R0            ;;POSITION FOR INDEXING
7074 040772 016000 041012      MOV    $TRPAD(R0),R0 ;;INDEX TO TABLE
7075 040776 000200      RTS     R0           ;;GO TO ROUTINE
7076
7077
7078      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
7079
7080 041000 011646      $TRAP2: MOV   (SP),-(SP) ;;MOVE THE PC DOWN
7081 041002 016666 000004 000002  MOV   4(SP),2(SP)    ;;MOVE THE PSW DOWN
7082 041010 000002      RTI                ;;RESTORE THE PSW
7083
7084      .SBTTL TRAP TABLE
7085
7086      ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
7087      ;*BY THE "TRAP" INSTRUCTION.
7088
7089      :      ROUTINE
7090      :      -----
7091 041012 041000      $TRPAD: .WORD  $TRAP2
7092 041014 036200      $TYPE   ;;CALL=TYPE   TRAP+1(104401) TTY TYPEOUT ROUTINE
7093 041016 036506      $TYPOC  ;;CALL=TYPOC  TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
7094 041020 036462      $TYPOS  ;;CALL=TYPOS  TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
7095 041022 036522      $TYPON  ;;CALL=TYPON  TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
7096 041024 036710      $TYPDS  ;;CALL=TYPDS  TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
7097
7098 041026 037522      $GTSWR  ;;CALL=GTSWR  TRAP+6(104406) GET SOFT-SWR SETTING
7099
7100 041030 037432      $CKSWR  ;;CALL=CKSWR  TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
7101 041032 037774      $RDCHR  ;;CALL=RDCHR  TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
7102 041034 040064      $RDLIN  ;;CALL=RDLIN  TRAP+11(104411) TTY TYPEIN STRING ROUTINE
7103 041036 040372      $RDOCT  ;;CALL=RDOCT  TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
7104 041040 040532      $SAVREG ;;CALL=SAVREG  TRAP+13(104413) SAVE R0-R5 ROUTINE
7105 041042 040570      $RESREG ;;CALL=RESREG  TRAP+14(104414) RESTORE R0-R5 ROUTINE
7106 041044 033676      $SCOP1$ ;;CALL=SCOP1$  TRAP+15(104415) INTERNAL LOOP ON ERROR
7107
      .SBTTL DATA TABLE FOR PRINT OUT

```

7108							
7109	041046	001220	003244		DT000:	.WORD	\$TESTN,TRAPPC
7110	041052	001220	001116	003224	DT002:	.WORD	\$TESTN,\$ERRPC,E.MR1,T.MR1,P1.BIT,PR.BIT,M1.BIT,M2.BIT,BITCNT
7111	041060	003164	003252	003254			
7112	041066	003256	003260	003262			
7113	041074	001220	001116	003200	DT003:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.MR2,T.MR2,E.MR3,T.MR3
7114	041102	003140	003226	003166			
7115	041110	003230	003170				
7116	041114	001220	001116	003200	DT041:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER,E.BA,T.BA,E.WC,T.WC
7117	041122	003140	003210	003150			
7118	041130	003214	003154	003204			
7119	041136	003144	003202	003142			
7120	041144	001220	001116	003200	DT046:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER
7121	041152	003140	003210	003150			
7122	041160	003214	003154				
7123	041164	001220	001116	003222	DT051:	.WORD	\$TESTN,\$ERRPC,E.DB,T.DB,WRDCNT
7124	041172	003162	003264				
7125	041176	001220	001116	003200	DT052:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,T.CS2,T.ER
7126	041204	003140	003150	003154			
7127	041212	001220	001116	003224	DT053:	.WORD	\$TESTN,\$ERRPC,E.MR1,T.MR1
7128	041220	003164					
7129	041222	001220	001116	003220	DT054:	.WORD	\$TESTN,\$ERRPC,E.DCYL,T.DCYL,E.DA,T.DA
7130	041230	003160	003206	003146			
7131	041236	001220	001116	003200	DT071:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER
7132	041244	003140	003210	003150			
7133	041252	003214	003154				
7134	041256	053564	053566	053570		.WORD	VRCHDR,VRCHDR+2,VRCHDR+4
7135	041264	001220	001116	003200	DT074:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER,HDRCNT
7136	041272	003140	003210	003150			
7137	041300	003214	003154	003310			
7138	041306	001220	001116	003224	DT077:	.WORD	\$TESTN,\$ERRPC,E.MR1,T.MR1,HDRCNT
7139	041314	003164	003310				
7140	041320	001220	001116	003234	DT134:	.WORD	\$TESTN,\$ERRPC,E.ECPT,T.ECPT,BITCNT
7141	041326	003174	003262				
7142	041332	001220	001116	003200	DT144:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER
7143	041340	003140	003210	003150			
7144	041346	003214	003154				
7145	041352	003204	003144	003202		.WORD	E.BA,T.BA,E.WC,T.WC,E.DCYL,T.DCYL,E.DA,T.DA
7146	041360	003142	003220	003160			
7147	041366	003206	003146				
7148	041372	001220	001116	003234	DT151:	.WORD	\$TESTN,\$ERRPC,E.ECPT,T.ECPT
7149	041400	003174					
7150						.SBTTL	DATA FORMAT FOR PRINT OUT
7151							
7152	041402	000001			DF000:	.WORD	1
7153	041404	002	000			.BYTE	2,0
7154	041406	000005			DF002:	.WORD	5
7155	041410	000	000			.BYTE	0,0
7156	041412	042455				.WORD	DH000A
7157	041414	000	000			.BYTE	0,0
7158	041416	042473				.WORD	DH000B
7159	041420	002	000			.BYTE	2,0
7160	041422	042537				.WORD	DH002A
7161	041424	000	000			.BYTE	0,0
7162	041426	042623				.WORD	DH002B
7163	041430	007	000			.BYTE	7,0

7164	041432	00005		DF003:	.WORD	5		:ERROR 3-24
7165	041434	000	000		.BYTE	0,0		
7166	041436	042455			.WORD	DH000A		
7167	041440	000	000		.BYTE	0,0		
7168	041442	042473			.WORD	DH000B		
7169	041444	002	000		.BYTE	2,0		
7170	041446	042711			.WORD	DH003A		
7171	041450	000	000		.BYTE	0,0		
7172	041452	042770			.WORD	DH003B		
7173	041454	006	000		.BYTE	6,0		
7174	041456	000005		DF025:	.WORD	5		:ERROR 25-40
7175	041460	000	000		.BYTE	0,0		
7176	041462	042455			.WORD	DH000A		
7177	041464	000	000		.BYTE	0,0		
7178	041466	042473			.WORD	DH000B		
7179	041470	002	000		.BYTE	2,0		
7180	041472	043047			.WORD	DH025A		
7181	041474	000	000		.BYTE	0,0		
7182	041476	043126			.WORD	DH025B		
7183	041500	006	000		.BYTE	6,0		
7184	041502	000007		DF041:	.WORD	7		
7185	041504	000	000		.BYTE	0,0		
7186	041506	042455			.WORD	DH000A		
7187	041510	000	000		.BYTE	0,0		
7188	041512	042473			.WORD	DH000B		
7189	041514	002	000		.BYTE	2,0		
7190	041516	043206			.WORD	DH041A		
7191	041520	000	000		.BYTE	0,0		
7192	041522	043265			.WORD	DH041B		
7193	041524	006	000		.BYTE	6,0		
7194	041526	043342			.WORD	DH041C		
7195	041530	000	000		.BYTE	0,0		
7196	041532	043401			.WORD	DH041D		
7197	041534	004	000		.BYTE	4,0		
7198	041536	000005		DF046:	.WORD	5		
7199	041540	000	000		.BYTE	0,0		
7200	041542	042455			.WORD	DH000A		
7201	041544	000	000		.BYTE	0,0		
7202	041546	042473			.WORD	DH000B		
7203	041550	002	000		.BYTE	2,0		
7204	041552	043206			.WORD	DH041A		
7205	041554	000	000		.BYTE	0,0		
7206	041556	043265			.WORD	DH041B		
7207	041560	006	000		.BYTE	6,0		
7208	041562	000005		DF051:	.WORD	5		
7209	041564	000	000		.BYTE	0,0		
7210	041566	042455			.WORD	DH000A		
7211	041570	000	000		.BYTE	0,0		
7212	041572	042473			.WORD	DH000B		
7213	041574	002	000		.BYTE	2,0		
7214	041576	043436			.WORD	DH051A		
7215	041600	000	000		.BYTE	0,0		
7216	041602	043463			.WORD	DH051B		
7217	041604	003	000		.BYTE	3,0		
7218	041606	000005		DF052:	.WORD	5		
7219	041610	000	000		.BYTE	0,0		

7220	041612	042455		.WORD	DH000A	
7221	041614	000	000	.BYTE	0,0	
7222	041616	042473		.WORD	DH000B	
7223	041620	002	000	.BYTE	2,0	
7224	041622	043511		.WORD	DH052A	
7225	041624	000	000	.BYTE	0,0	
7226	041626	043530		.WORD	DH052B	
7227	041630	004	000	.BYTE	4,0	
7228	041632	000005		.WORD	5	
7229	041634	000	000	.BYTE	0,0	
7230	041636	042455		.WORD	DH000A	
7231	041640	000	000	.BYTE	0,0	
7232	041642	042473		.WORD	DH000B	
7233	041644	002	000	.BYTE	2,0	
7234	041646	043565		.WORD	DH053A	
7235	041650	000	000	.BYTE	0,0	
7236	041652	043604		.WORD	DH053B	
7237	041654	002	000	.BYTE	2,0	
7238	041656	000005		.WORD	5	:ERROR-54-61
7239	041660	000	000	.BYTE	0,0	
7240	041662	042455		.WORD	DH000A	
7241	041664	000	000	.BYTE	0,0	
7242	041666	042473		.WORD	DH000B	
7243	041670	002	000	.BYTE	2,0	
7244	041672	043622		.WORD	DH054A	
7245	041674	000	000	.BYTE	0,0	
7246	041676	043661		.WORD	DH054B	
7247	041700	004	000	.BYTE	4,0	
7248	041702	000007		.WORD	7	:ERRORS 71-73
7249	041704	000	000	.BYTE	0,0	
7250	041706	042455		.WORD	DH000A	
7251	041710	000	000	.BYTE	0,0	
7252	041712	042473		.WORD	DH000B	
7253	041714	002	000	.BYTE	2,0	
7254	041716	043206		.WORD	DH041A	
7255	041720	000	000	.BYTE	0,0	
7256	041722	043265		.WORD	DH041B	
7257	041724	006	000	.BYTE	6,0	
7258	041726	043716		.WORD	DH071A	
7259	041730	000	000	.BYTE	0,0	
7260	041732	043737		.WORD	DH071B	
7261	041734	003	000	.BYTE	3,0	
7262	041736	000005		.WORD	5	
7263	041740	000	000	.BYTE	0,0	
7264	041742	042455		.WORD	DH000A	
7265	041744	000	000	.BYTE	0,0	
7266	041746	042473		.WORD	DH000B	
7267	041750	002	000	.BYTE	2,0	
7268	041752	043766		.WORD	DH074A	
7269	041754	000	000	.BYTE	0,0	
7270	041756	044055		.WORD	DH074B	
7271	041760	007	000	.BYTE	7,0	
7272	041762	000005		.WORD	5	
7273	041764	000	000	.BYTE	0,0	
7274	041766	042455		.WORD	DH000A	
7275	041770	000	000	.BYTE	0,0	

7276	041772	042473				.WORD	DH000B
7277	041774	002	000			.BYTE	2,0
7278	041776	044141				.WORD	DH077A
7279	042000	000	000			.BYTE	0,0
7280	042002	044170				.WORD	DH077B
7281	042004	003	000			.BYTE	3,0
7282	042006	000005		DF134:		.WORD	5
7283	042010	000	000			.BYTE	0,0
7284	042012	042455				.WORD	DH000A
7285	042014	000	000			.BYTE	0,0
7286	042016	042473				.WORD	DH000B
7287	042020	002	000			.BYTE	2,0
7288	042022	044214				.WORD	DH134A
7289	042024	000	000			.BYTE	0,0
7290	042026	044240				.WORD	DH134B
7291	042030	003	000			.BYTE	3,0
7292	042032	000007		DF144:		.WORD	7
7293	042034	000	000			.BYTE	0,0
7294	042036	042455				.WORD	DH000A
7295	042040	000	000			.BYTE	0,0
7296	042042	042473				.WORD	DH000B
7297	042044	002	000			.BYTE	2,0
7298	042046	043206				.WORD	DH041A
7299	042050	000	000			.BYTE	0,0
7300	042052	043265				.WORD	DH041B
7301	042054	006	000			.BYTE	6,0
7302	042056	044266				.WORD	DH144A
7303	042060	000	000			.BYTE	0,0
7304	042062	044365				.WORD	DH144B
7305	042064	010	000			.BYTE	10,0
7306	042066	000005		DF151:		.WORD	5
7307	042070	000	000			.BYTE	0,0
7308	042072	042455				.WORD	DH000A
7309	042074	000	000			.BYTE	0,0
7310	042076	042473				.WORD	DH000B
7311	042100	002	000			.BYTE	2,0
7312	042102	043511				.WORD	DH052A
7313	042104	000	000			.BYTE	0,0
7314	042106	044462				.WORD	DH151A
7315	042110	002	000			.BYTE	2,0
7316						.SBTTL	ASCII MESSAGES
7317							
7318	042112	005015	045522	030466	OPR001:	.ASCIZ	<15><12>/RK611 BUS ADDRESS ( /
7319	042120	020061	052502	020123			
7320	042126	042101	051104	051505			
7321	042134	020123	020050	000			
7322	042141	040	020051	020075	OPR002:	.ASCIZ	/ ) = /
7323	042146	000					
7324	042147	122	033113	030461	OPR003:	.ASCIZ	/RK611 VECTOR ADDRESS ( /
7325	042154	053040	041505	047524			
7326	042162	020122	042101	051104			
7327	042170	051505	020123	020050			
7328	042176	000					
7329	042177	122	033113	030461	OPR004:	.ASCIZ	/RK611 PRIORITY ( /
7330	042204	050040	044522	051117			
7331	042212	052111	020131	020050			

```
7332 042220 000
7333 042221 015 025012 025052 OPR005: .ASCIZ <15><12>/***** PROGRAM HALTED *****/<15><12>
7334 042226 025052 020040 051120
7335 042234 043517 040522 020115
7336 042242 040510 052114 042105
7337 042250 025040 025052 025052
7338 042256 005015 000
7339 042261 015 051412 041505 OPR006: .ASCIZ <15><12>/SECOND PASS RUN TIME IS APPROX 3:15 MINUTES/<15><12>
7340 042266 047117 020104 040520
7341 042274 051523 051040 047125
7342 042302 052040 046511 020105
7343 042310 051511 040440 050120
7344 042316 047522 020130 035063
7345 042324 032461 046440 047111
7346 042332 052125 051505 005015
7347 042340 000
7348 042341 040 000040 SPACE2: .ASCIZ / /
7349 042344 005015 051120 043517 ABORT: .ASCIZ <15><12>/PROGRAM ABORTED BECAUSE ERROR THRESHOLD EXCEEDED/<15><12>
7350 042352 040522 020115 041101
7351 042360 051117 042524 020104
7352 042366 042502 040503 051525
7353 042374 020105 051105 047522
7354 042402 020122 044124 042522
7355 042410 044123 046117 020104
7356 042416 054105 042503 042105
7357 042424 042105 005015 000
7358 042431 015 052012 051505 TSTBY1: .ASCIZ <15><12>/TEST /
7359 042436 020124 000
7360 042441 040 054502 040520 TSTBY2: .ASCIZ / BYPASSED/<15><12>
7361 042446 051523 042105 005015
7362 042454 000
7363 .SBTTL DATA HEADERS
7364
7365 042455 124 051505 020124 DH000A: .ASCIZ /TEST ERROR/
7366 042462 020040 042440 051122
7367 042470 051117 000
7368 042473 116 046525 020040 DH000B: .ASCIZ /NUM PC/
7369 042500 020040 050040 000103
7370 042506 042524 052123 020040 DH000C: .ASCII /TEST TRAP/<15><12>
7371 042514 020040 051124 050101
7372 042522 005015
7373 042524 052516 020115 020040 .ASCIZ /NUM PC/
7374 042532 020040 041520 000
7375 042537 105 050130 041505 DH002A: .ASCIZ /EXPECT ACTUAL PRESENT PRESENT PRESENT PRESENT BIT/
7376 042544 020124 040440 052103
7377 042552 040525 020114 050040
7378 042560 042522 042523 052116
7379 042566 050040 042522 042523
7380 042574 052116 050040 042522
7381 042602 042523 052116 050040
7382 042610 042522 042523 052116
7383 042616 041040 052111 000
7384 042623 122 046513 030522 DH002B: .ASCIZ /RKMR1 RKMR1 BIT+1 BIT BIT-1 BIT-2 COUNT/
7385 042630 020040 051040 046513
7386 042636 030522 020040 041040
7387 042644 052111 030453 020040
```

7388 042652 041040 052111 020040  
7389 042660 020040 041040 052111  
7390 042666 030455 020040 041040  
7391 042674 052111 031055 020040  
7392 042702 041440 052517 052116  
7393 042710 000  
7394 042711 105 050130 041505  
7395 042716 020124 040440 052103  
7396 042724 040525 020114 042440  
7397 042732 050130 041505 020124  
7398 042740 040440 052103 040525  
7399 042746 020114 042440 050130  
7400 042754 041505 020124 040440  
7401 042762 052103 040525 000114  
7402 042770 045522 051503 020061  
7403 042776 020040 045522 051503  
7404 043004 020061 020040 042515  
7405 043012 051523 040440 020040  
7406 043020 042515 051523 040440  
7407 043026 020040 042515 051523  
7408 043034 041040 020040 042515  
7409 043042 051523 041040 000  
7410 043047 105 050130 041505  
7411 043054 020124 040440 052103  
7412 043062 040525 020114 042440  
7413 043070 050130 041505 020124  
7414 043076 040440 052103 040525  
7415 043104 020114 042440 050130  
7416 043112 041505 020124 040440  
7417 043120 052103 040525 000114  
7418 043126 045522 051503 020061  
7419 043134 020040 045522 051503  
7420 043142 020061 020040 042110  
7421 043150 053440 020104 020061  
7422 043156 042110 053440 020104  
7423 043164 020061 042110 053440  
7424 043172 020104 020062 042110  
7425 043200 053440 020104 000062  
7426 043206 054105 042520 052103  
7427 043214 020040 041501 052524  
7428 043222 046101 020040 054105  
7429 043230 042520 052103 020040  
7430 043236 041501 052524 046101  
7431 043244 020040 054105 042520  
7432 043252 052103 020040 041501  
7433 043260 052524 046101 000  
7434 043265 122 041513 030523  
7435 043272 020040 051040 041513  
7436 043300 030523 020040 051040  
7437 043306 041513 031123 020040  
7438 043314 051040 041513 031123  
7439 043322 020040 051040 042513  
7440 043330 020122 020040 051040  
7441 043336 042513 000122  
7442 043342 054105 042520 052103  
7443 043350 020040 041501 052524

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/

DH041A: .ASCIZ /EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL/

DH041B: .ASCIZ /RKCS1 RKCS1 RKCS2 RKCS2 RKER RKER/

DH041C: .ASCIZ /EXPECT ACTUAL EXPECT ACTUAL/



G 11

7444	043356	046101	020040	054105					
7445	043364	042520	052103	020040					
7446	043372	041501	052524	046101					
7447	043400	000							
7448	043401	122	041113	020101	DH041D:	.ASCIZ	/RKBA	RKBA	RKWC RKWC/
7449	043406	020040	051040	041113					
7450	043414	020101	020040	051040					
7451	043422	053513	020103	020040					
7452	043430	051040	053513	000103					
7453	043436	040504	040524	020040	DH051A:	.ASCIZ	/DATA	DATA	WORD/
7454	043444	020040	040504	040524					
7455	043452	020040	020040	047527					
7456	043460	042122	000						
7457	043463	105	050130	041505	DH051B:	.ASCIZ	/EXPECT	READ	COUNT/
7458	043470	020124	051040	040505					
7459	043476	020104	020040	041440					
7460	043504	052517	052116	000					
7461	043511	105	050130	041505	DH052A:	.ASCIZ	/EXPECT	ACTUAL/	
7462	043516	020124	040440	052103					
7463	043524	040525	000114						
7464	043530	045522	051503	020061	DH052B:	.ASCIZ	/RKCS1	RKCS1	RKCS2 RKER/
7465	043536	020040	045522	051503					
7466	043544	020061	020040	045522					
7467	043552	051503	020062	020040					
7468	043560	045522	051105	000					
7469	043565	105	050130	041505	DH053A:	.ASCIZ	/EXPECT	ACTUAL/	
7470	043572	020124	040440	052103					
7471	043600	040525	000114						
7472	043604	045522	051115	020061	DH053B:	.ASCIZ	/RKMR1	RKMR1/	
7473	043612	020040	045522	051115					
7474	043620	000061							
7475	043622	054105	042520	052103	DH054A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT ACTUAL/
7476	043630	020040	041501	052524					
7477	043636	046101	020040	054105					
7478	043644	042520	052103	020040					
7479	043652	041501	052524	046101					
7480	043660	000							
7481	043661	122	042113	054503	DH054B:	.ASCIZ	/RKDCYL	RKDCYL	RKDA RKDA/
7482	043666	020114	051040	042113					
7483	043674	054503	020114	051040					
7484	043702	042113	020101	020040					
7485	043710	051040	042113	000101					
7486	043716	042510	042101	051105	DH071A:	.ASCIZ	/HEADER	SIMULATED/	
7487	043724	051440	046511	046125					
7488	043732	052101	042105	000					
7489	043737	127	051117	020104	DH071B:	.ASCIZ	/WORD 1	WORD 2	WORD 3/
7490	043744	020061	053440	051117					
7491	043752	020104	020062	053440					
7492	043760	051117	020104	000063					
7493	043766	054105	042520	052103	DH074A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT ACTUAL EXPECT ACTUAL HEADER/
7494	043774	020040	041501	052524					
7495	044002	046101	020040	054105					
7496	044010	042520	052103	020040					
7497	044016	041501	052524	046101					
7498	044024	020040	054105	042520					
7499	044032	052103	020040	041501					

7500	044040	052524	046101	020040
7501	044046	042510	042101	051105
7502	044054	000		
7503	044055	122	041513	030523
7504	044062	020040	051040	041513
7505	044070	030523	020040	051040
7506	044076	041513	031123	020040
7507	044104	051040	041513	031123
7508	044112	020040	051040	042513
7509	044120	020122	020040	051040
7510	044126	042513	020122	020040
7511	044134	047040	046525	000
7512	044141	105	050130	041505
7513	044146	020124	040440	052103
7514	044154	040525	020114	044040
7515	044162	040505	042504	000122
7516	044170	045522	051115	020061
7517	044176	020040	045522	051115
7518	044204	020061	020040	052516
7519	044212	000115		
7520	044214	054105	042520	052103
7521	044222	020040	041501	052524
7522	044230	046101	020040	044502
7523	044236	000124		
7524	044240	045522	041505	052120
7525	044246	020040	045522	041505
7526	044254	052120	020040	047503
7527	044262	047125	000124	
7528	044266	054105	042520	052103
7529	044274	020040	041501	052524
7530	044302	046101	020040	054105
7531	044310	042520	052103	020040
7532	044316	041501	052524	046101
7533	044324	020040	054105	042520
7534	044332	052103	020040	041501
7535	044340	052524	046101	020040
7536	044346	054105	042520	052103
7537	044354	020040	041501	052524
7538	044362	046101	000	
7539	044365	122	041113	020101
7540	044372	020040	051040	041113
7541	044400	020101	020040	051040
7542	044406	053513	020103	020040
7543	044414	051040	053513	020103
7544	044422	020040	051040	042113
7545	044430	054503	020114	051040
7546	044436	042113	054503	020114
7547	044444	051040	042113	020101
7548	044452	020040	051040	042113
7549	044460	000101		
7550	044462	045522	041505	052120
7551	044470	020040	045522	041505
7552	044476	052120	000	
7553				
7554				
7555	044501	125	042516	050130

DH074B: .ASCIZ /RKCS1 RKCS1 RKCS2 RKCS2 RKER RKER NUM/

DH077A: .ASCIZ /EXPECT ACTUAL HEADER/

DH077B: .ASCIZ /RKMR1 RKMR1 NUM/

DH134A: .ASCIZ /EXPECT ACTUAL BIT/

DH134B: .ASCIZ /RKECPT RKECPT COUNT/

DH144A: .ASCIZ /EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL/

DH144B: .ASCIZ /RKBA RKBA RKWC RKWC RKDCYL RKDCYL RKDA RKDA/

DH151A: .ASCIZ /RKECPT RKECPT/

.SBTTL ERROR MESSAGES

EM000: .ASCIZ /UNEXPECTED MEMORY PARITY ENABLE TRAP/

7556	044506	041505	042524	020104	
7557	044514	042515	047515	054522	
7558	044522	050040	051101	052111	
7559	044530	020131	047105	041101	
7560	044536	042514	052040	040522	
7561	044544	000120			
7562	044546	052101	042524	050115	EM300: .ASCIZ /ATTEMPTING TO CHECK SEEK MESS FROM READ DATA/
7563	044554	044524	043516	052040	
7564	044562	020117	044103	041505	
7565	044570	020113	042523	045505	
7566	044576	046440	051505	020123	
7567	044604	051106	046517	051040	
7568	044612	040505	020104	040504	
7569	044620	040524	000		
7570	044623	101	052124	046505	EM301: .ASCIZ /ATTEMPTING TO CHECK SEEK MESS FROM WRITE DATA/
7571	044630	052120	047111	020107	
7572	044636	047524	041440	042510	
7573	044644	045503	051440	042505	
7574	044652	020113	042515	051523	
7575	044660	043040	047522	020115	
7576	044666	051127	052111	020105	
7577	044674	040504	040524	000	
7578	044701	101	052124	046505	EM302: .ASCIZ /ATTEMPTING TO CHECK SEEK MESS FROM WRITE CHECK/
7579	044706	052120	047111	020107	
7580	044714	047524	041440	042510	
7581	044722	045503	051440	042505	
7582	044730	020113	042515	051523	
7583	044736	043040	047522	020115	
7584	044744	051127	052111	020105	
7585	044752	044103	041505	000113	
7586	044760	052101	042524	050115	EM303: .ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM READ DATA/
7587	044766	044524	043516	052040	
7588	044774	020117	044103	041505	
7589	045002	020113	046103	040505	
7590	045010	020122	042515	051523	
7591	045016	043040	047522	020115	
7592	045024	042522	042101	042040	
7593	045032	052101	000101		
7594	045036	052101	042524	050115	EM304: .ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM WRITE DATA/
7595	045044	044524	043516	052040	
7596	045052	020117	044103	041505	
7597	045060	020113	046103	040505	
7598	045066	020122	042515	051523	
7599	045074	043040	047522	020115	
7600	045102	051127	052111	020105	
7601	045110	040504	040524	000	
7602	045115	101	052124	046505	EM305: .ASCIZ /ATTEMPTING TO CHECK CLEAR MESS FROM WRITE CHECK/
7603	045122	052120	047111	020107	
7604	045130	047524	041440	042510	
7605	045136	045503	041440	042514	
7606	045144	051101	046440	051505	
7607	045152	020123	051106	046517	
7608	045160	053440	044522	042524	
7609	045166	041440	042510	045503	
7610	045174	000			
7611	045175	101	052124	046505	EM306: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>

7612	045202	052120	047111	020107
7613	045210	047524	041440	042510
7614	045216	045503	044040	040505
7615	045224	020104	042507	042516
7616	045232	040522	044524	047117
7617	045240	005015		
7618	045242	044527	044124	053040
7619	045250	051101	047511	051525
7620	045256	041440	046131	047111
7621	045264	042504	020122	040526
7622	045272	052514	051505	000
7623	045277	101	052124	046505
7624	045304	052120	047111	020107
7625	045312	047524	041440	042510
7626	045320	045503	044040	040505
7627	045326	020104	042507	042516
7628	045334	040522	044524	047117
7629	045342	005015		
7630	045344	044527	044124	053040
7631	045352	051101	047511	051525
7632	045360	052040	040522	045503
7633	045366	053040	046101	042525
7634	045374	000123		
7635	045376	052101	042524	050115
7636	045404	044524	043516	052040
7637	045412	020117	044103	041505
7638	045420	020113	042510	042101
7639	045426	043440	047105	051105
7640	045434	052101	047511	006516
7641	045442	012		
7642	045443	127	052111	020110
7643	045450	040526	044522	052517
7644	045456	020123	042523	052103
7645	045464	051117	053040	046101
7646	045472	042525	000123	
7647	045476	052101	042524	050115
7648	045504	044524	043516	052040
7649	045512	020117	044103	041505
7650	045520	020113	042510	042101
7651	045526	043440	047105	051105
7652	045534	052101	047511	006516
7653	045542	012		
7654	045543	127	052111	020110
7655	045550	040526	044522	052517
7656	045556	020123	047506	046522
7657	045564	052101	053040	046101
7658	045572	042525	000123	
7659	045576	052101	042524	050115
7660	045604	044524	043516	052040
7661	045612	020117	044103	041505
7662	045620	020113	050116	020122
7663	045626	040504	040524	052040
7664	045634	040522	051516	042506
7665	045642	020122	047506	020122
7666	045650	051127	052111	020105
7667	045656	040504	040524	000

.ASCIZ /WITH VARIOUS CYLINDER VALUES/

EM307: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>

.ASCIZ /WITH VARIOUS TRACK VALUES/

EM308: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>

.ASCIZ /WITH VARIOUS SECTOR VALUES/

EM309: .ASCII /ATTEMPTING TO CHECK HEAD GENERATION/<15><12>

.ASCIZ /WITH VARIOUS FORMAT VALUES/

EM310: .ASCIZ /ATTEMPTING TO CHECK NPR DATA TRANSFER FOR WRITE DATA/

7668	045663	101	052124	046505	EM311: .ASCIZ /ATTEMPTING TO CHECK HEADER RECOGNITION/
7669	045670	052120	047111	020107	
7670	045676	047524	041440	042510	
7671	045704	045503	044040	040505	
7672	045712	042504	020122	042522	
7673	045720	047503	047107	052111	
7674	045726	047511	000116		
7675	045732	052101	042524	050115	EM312: .ASCIZ /ATTEMPTING TO CHECK SECTOR INCREMENT/
7676	045740	044524	043516	052040	
7677	045746	020117	044103	041505	
7678	045754	020113	042523	052103	
7679	045762	051117	044440	041516	
7680	045770	042522	042515	052116	
7681	045776	000			
7682	045777	101	052124	046505	EM313: .ASCIZ /ATTEMPTING TO CHECK TRACK INCREMENT/
7683	046004	052120	047111	020107	
7684	046012	047524	041440	042510	
7685	046020	045503	052040	040522	
7686	046026	045503	044440	041516	
7687	046034	042522	042515	052116	
7688	046042	000			
7689	046043	101	052124	046505	EM314: .ASCIZ /ATTEMPTING TO CHECK CYLINDER INCREMENT/
7690	046050	052120	047111	020107	
7691	046056	047524	041440	042510	
7692	046064	045503	041440	046131	
7693	046072	047111	042504	020122	
7694	046100	047111	051103	046505	
7695	046106	047105	000124		
7696	046112	052101	042524	050115	EM315: .ASCII /ATTEMPTING TO CHECK SECTOR PULSE DETECTION/<15><12>
7697	046120	044524	043516	052040	
7698	046126	020117	044103	041505	
7699	046134	020113	042523	052103	
7700	046142	051117	050040	046125	
7701	046150	042523	042040	052105	
7702	046156	041505	044524	047117	
7703	046164	005115			
7704	046166	044527	044124	053440	.ASCIZ /WITH WRITE DATA/
7705	046174	044522	042524	042040	
7706	046202	052101	000101		
7707	046206	052101	042524	050115	EM316: .ASCIZ /ATTEMPTING TO FORCE BAD SECTOR ERROR/
7708	046214	044524	043516	052040	
7709	046222	020117	047506	041522	
7710	046230	020105	040502	020104	
7711	046236	042523	052103	051117	
7712	046244	042440	051122	051117	
7713	046252	000			
7714	046253	101	052124	046505	EM317: .ASCII /ATTEMPTING TO FORCE HEADER VRC ERROR/<15><12>
7715	046260	052120	047111	020107	
7716	046266	047524	043040	051117	
7717	046274	042503	044040	040505	
7718	046302	042504	020122	051126	
7719	046310	020103	051105	047522	
7720	046316	006522	012		
7721	046321	127	052111	020110	.ASCIZ /WITH BAD SECTOR PRESENT/
7722	046326	040502	020104	042523	
7723	046334	052103	051117	050040	

7724	046342	042522	042523	052116	
7725	046350	000			
7726	046351	101	052124	046505	EM318: .ASCIZ /ATTEMPTING TO FORCE HVRC ERROR/
7727	046356	052120	047111	020107	
7728	046364	047524	043040	051117	
7729	046372	042503	044040	051126	
7730	046400	020103	051105	047522	
7731	046406	000122			
7732	046410	052101	042524	050115	EM319: .ASCIZ /ATTEMPTING TO FORCE OPERATION INCOMPLETE/
7733	046416	044524	043516	052040	
7734	046424	020117	047506	041522	
7735	046432	020105	050117	051105	
7736	046440	052101	047511	020116	
7737	046446	047111	047503	050115	
7738	046454	042514	042524	000	
7739	046461	101	052124	046505	EM320: .ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 57/
7740	046466	044520	043516	052040	
7741	046474	020117	047506	041522	
7742	046502	020105	050117	020111	
7743	046510	044527	044124	044040	
7744	046516	051126	020103	051105	
7745	046524	047522	020122	047117	
7746	046532	044040	040505	042504	
7747	046540	020122	033463	000	
7748	046545	101	052124	046505	EM321: .ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 36/
7749	046552	044520	043516	052040	
7750	046560	020117	047506	041522	
7751	046566	020105	050117	020111	
7752	046574	044527	044124	044040	
7753	046602	051126	020103	051105	
7754	046610	047522	020122	047117	
7755	046616	044040	040505	042504	
7756	046624	020122	033063	000	
7757	046631	101	052124	046505	EM322: .ASCIZ /ATTEMPTING TO FORCE OPI WITH HVRC ERROR ON HEADER 0/
7758	046636	044520	043516	052040	
7759	046644	020117	047506	041522	
7760	046652	020105	050117	020111	
7761	046660	044527	044124	044040	
7762	046666	051126	020103	051105	
7763	046674	047522	020122	047117	
7764	046702	044040	040505	042504	
7765	046710	020122	000060		
7766	046714	044103	041505	044513	EM323: .ASCIZ /CHECKING HEADER RECOGNITION WITH PREVIOUS BAD SECTOR ERROR/
7767	046722	043516	044040	040505	
7768	046730	042504	020122	042522	
7769	046736	047503	047107	052111	
7770	046744	047511	020116	044527	
7771	046752	044124	050040	042522	
7772	046760	044526	052517	020123	
7773	046766	040502	020104	042523	
7774	046774	052103	051117	042440	
7775	047002	051122	051117	000	
7776	047007	103	042510	045503	EM324: .ASCIZ /CHECKING HEADER RECOGNITION WITH PREVIOUS HEADER VRC ERROR/
7777	047014	047111	020107	042510	
7778	047022	042101	051105	051040	
7779	047030	041505	043517	044516	

7780	047036	044524	047117	053440	
7781	047044	052111	020110	051120	
7782	047052	053105	047511	051525	
7783	047060	044040	040505	042504	
7784	047066	020122	051126	020103	
7785	047074	051105	047522	000122	
7786	047102	047506	041522	047111	EM325: .ASCIZ /FORCING BAD SECTOR ERROR WITH PREVIOUS HEADER VRC ERROR/
7787	047110	020107	040502	020104	
7788	047116	042523	052103	051117	
7789	047124	042440	051122	051117	
7790	047132	053440	052111	020110	
7791	047140	051120	053105	047511	
7792	047146	051525	044040	040505	
7793	047154	042504	020122	051126	
7794	047162	020103	051105	047522	
7795	047170	000122			
7796	047172	047506	041522	047111	EM326: .ASCIZ /FORCING HEAD VRC ERROR WITH PREVIOUS BAD SECTOR ERROR/
7797	047200	020107	042510	042101	
7798	047206	053040	041522	042440	
7799	047214	051122	051117	053440	
7800	047222	052111	020110	051120	
7801	047230	053105	047511	051525	
7802	047236	041040	042101	051440	
7803	047244	041505	047524	020122	
7804	047252	051105	047522	000122	
7805	047260	052101	042524	050115	EM327: .ASCIZ /ATTEMPTING TO WRITE SYNCH WITH WRITE DATA/
7806	047266	044524	043516	052040	
7807	047274	020117	051127	052111	
7808	047302	020105	054523	041516	
7809	047310	020110	044527	044124	
7810	047316	053440	044522	042524	
7811	047324	042040	052101	000101	
7812	047332	052101	042524	050115	EM328: .ASCIZ /ATTEMPTING TO WRITE DATA FIELD WITH WRITE DATA/
7813	047340	044524	043516	052040	
7814	047346	020117	051127	052111	
7815	047354	020105	040504	040524	
7816	047362	043040	042511	042114	
7817	047370	053440	052111	020110	
7818	047376	051127	052111	020105	
7819	047404	040504	040524	000	
7820	047411	101	052124	046505	EM329: .ASCIZ /ATTEMPTING TO CHECK ECC PATTERN CLEARING/
7821	047416	052120	047111	020107	
7822	047424	047524	041440	042510	
7823	047432	045503	042440	041503	
7824	047440	050040	052101	042524	
7825	047446	047122	041440	042514	
7826	047454	051101	047111	000107	
7827	047462	052101	042524	050115	EM330: .ASCIZ /ATTEMPTING TO CHECK ECC GENERATION/
7828	047470	044524	043516	052040	
7829	047476	020117	044103	041505	
7830	047504	020113	041505	020103	
7831	047512	042507	042516	040522	
7832	047520	044524	047117	000	
7833	047525	101	052124	046505	EM331: .ASCIZ /ATTEMPTING BAD SECTOR ERROR SETTING CONTROLLER ERROR/
7834	047532	052120	047111	020107	
7835	047540	040502	020104	042523	

7836	047546	052103	051117	042440
7837	047554	051122	051117	051440
7838	047562	052105	044524	043516
7839	047570	041440	047117	051124
7840	047576	046117	042514	020122
7841	047604	051105	047522	000122
7842	047612	052101	042524	050115
7843	047620	044524	043516	044040
7844	047626	040505	042504	020122
7845	047634	051126	020103	051105
7846	047642	047522	020122	042523
7847	047650	052124	047111	020107
7848	047656	047503	052116	047522
7849	047664	046114	051105	042440
7850	047672	051122	051117	000
7851	047677	101	052124	046505
7852	047704	052120	047111	020107
7853	047712	047524	053440	044522
7854	047720	042524	042440	041503
7855	047726	053440	051117	051504
7856	047734	000		
7857	047735	101	052124	046505
7858	047742	052120	047111	020107
7859	047750	047524	053440	044522
7860	047756	042524	050040	051517
7861	047764	040524	041115	042514
7862	047772	000		
7863	047773	101	052124	046505
7864	050000	052120	047111	020107
7865	050006	047503	050115	042514
7866	050014	042524	042440	042530
7867	050022	052503	044524	047117
7868	050030	047440	020106	051127
7869	050036	052111	020105	040504
7870	050044	040524	000	
7871	050047	101	052124	046505
7872	050054	052120	047111	020107
7873	050062	051105	047522	020122
7874	050070	046103	040505	020122
7875	050076	044527	044124	041440
7876	050104	047117	051124	046117
7877	050112	042514	020122	046103
7878	050120	040505	000122	
7879	050124	052101	042524	050115
7880	050132	044524	043516	050040
7881	050140	051101	044524	046101
7882	050146	051440	041505	047524
7883	050154	020122	051127	052111
7884	050162	020105	044527	044124
7885	050170	055040	051105	020117
7886	050176	044506	046114	000
7887	050203	101	052124	046505
7888	050210	052120	047111	020107
7889	050216	047524	053440	044522
7890	050224	042524	030440	020070
7891	050232	044502	020124	040504

EM332: .ASCIZ /ATTEMPTING HEADER VRC ERROR SETTING CONTROLLER ERROR/

EM333: .ASCIZ /ATTEMPTING TO WRITE ECC WORDS/

EM334: .ASCIZ /ATTEMPTING TO WRITE POSTAMBLE/

EM335: .ASCIZ /ATTEMPTING COMPLETE EXECUTION OF WRITE DATA/

EM336: .ASCIZ /ATTEMPTING ERROR CLEAR WITH CONTROLLER CLEAR/

EM337: .ASCIZ /ATTEMPTING PARTIAL SECTOR WRITE WITH ZERO FILL/

EM338: .ASCIZ /ATTEMPTING TO WRITE 18 BIT DATA FIELD WITH WRITE DATA/



7892	050240	040524	043040	042511	
7893	050246	042114	053440	052111	
7894	050254	020110	051127	052111	
7895	050262	020105	040504	040524	
7896	050270	000			
7897	050271	101	052124	046505	EM339: .ASCII /ATTEMPTING TO WRITE BIT 16-17 OF 18 BIT/
7898	050276	052120	047111	020107	
7899	050304	047524	053440	044522	
7900	050312	042524	041040	052111	
7901	050320	030440	026466	033461	
7902	050326	047440	020106	034061	
7903	050334	041040	052111		
7904	050340	005015	040504	040524	.ASCIZ <15><12>/DATA FIELD WITH WRITE DATA/
7905	050346	043040	042511	042114	
7906	050354	053440	052111	020110	
7907	050362	051127	052111	020105	
7908	050370	040504	040524	000	
7909	050375	101	052124	046505	EM340: .ASCIZ /ATTEMPTING WRITE DATA IN 18 BIT MODE/
7910	050402	052120	047111	020107	
7911	050410	051127	052111	020105	
7912	050416	040504	040524	044440	
7913	050424	020116	034061	041040	
7914	050432	052111	046440	042117	
7915	050440	000105			
7916	050442	051503	020061	047111	EM4000: .ASCIZ /CS1 INCORRECT/
7917	050450	047503	051122	041505	
7918	050456	000124			
7919	050460	042515	051523	040440	EM4001: .ASCIZ /MESS A INCORRECT/
7920	050466	044440	041516	051117	
7921	050474	042522	052103	000	
7922	050501	115	051505	020123	EM4002: .ASCIZ /MESS B INCORRECT/
7923	050506	020102	047111	047503	
7924	050514	051122	041505	000124	
7925	050522	042510	042101	051105	EM4003: .ASCIZ /HEADER WORD 1 INCORRECT/
7926	050530	053440	051117	020104	
7927	050536	020061	047111	047503	
7928	050544	051122	041505	000124	
7929	050552	042510	042101	051105	EM4004: .ASCIZ /HEADER WORD 2 INCORRECT/
7930	050560	053440	051117	020104	
7931	050566	020062	047111	047503	
7932	050574	051122	041505	000124	
7933	050602	051503	020062	047111	EM4005: .ASCIZ /CS2 INCORRECT/
7934	050610	047503	051122	041505	
7935	050616	000124			
7936	050620	051105	047522	020122	EM4006: .ASCIZ /ERROR REG INCORRECT/
7937	050626	042522	020107	047111	
7938	050634	047503	051122	041505	
7939	050642	000124			
7940	050644	052502	020123	042101	EM4007: .ASCIZ /BUS ADDRESS INCORRECT/
7941	050652	051104	051505	020123	
7942	050660	047111	047503	051122	
7943	050666	041505	000124		
7944	050672	047527	042122	041440	EM4008: .ASCIZ /WORD COUNT INCORRECT/
7945	050700	052517	052116	044440	
7946	050706	041516	051117	042522	
7947	050714	052103	000		

7948	050717	103	030523	044440	EM4009: .ASCIZ /CS1 INCORRECT AFTER READING DATA BUFFER/
7949	050724	041516	051117	042522	
7950	050732	052103	040440	052106	
7951	050740	051105	051040	040505	
7952	050746	044504	043516	042040	
7953	050754	052101	020101	052502	
7954	050762	043106	051105	000	
7955	050767	103	031123	044440	EM4010: .ASCIZ /CS2 INCORRECT AFTER READING DATA BUFFER/
7956	050774	041516	051117	042522	
7957	051002	052103	040440	052106	
7958	051010	051105	051040	040505	
7959	051016	044504	043516	042040	
7960	051024	052101	020101	052502	
7961	051032	043106	051105	000	
7962	051037	105	051122	051117	EM4011: .ASCIZ /ERROR REG INCORRECT AFTER READING DATA BUFFER/
7963	051044	051040	043505	044440	
7964	051052	041516	051117	042522	
7965	051060	052103	040440	052106	
7966	051066	051105	051040	040505	
7967	051074	044504	043' 16	042040	
7968	051102	052101	020101	052502	
7969	051110	043106	051105	000	
7970	051115	104	052101	020101	EM4012: .ASCIZ /DATA READ FROM MEMORY INCORRECT/
7971	051122	042522	042101	043040	
7972	051130	047522	020115	042515	
7973	051136	047515	054522	044440	
7974	051144	041516	051117	042522	
7975	051152	052103	000		
7976	051155	115	030522	044440	EM4013: .ASCIZ /MR1 INCORRECT AFTER GAP IN WRITE DATA/
7977	051162	041516	051117	042522	
7978	051170	052103	040440	052106	
7979	051176	051105	043440	050101	
7980	051204	044440	020116	051127	
7981	051212	052111	020105	040504	
7982	051220	040524	000		
7983	051223	104	051511	020113	EM4014: .ASCIZ /DISK ADDRESS REG. INCORRECT/
7984	051230	042101	051104	051505	
7985	051236	020123	042522	027107	
7986	051244	044440	041516	051117	
7987	051252	042522	052103	000	
7988	051257	103	046131	047111	EM4015: .ASCIZ /CYLINDER ADDRESS REG. INCORRECT/
7989	051264	042504	020122	042101	
7990	051272	051104	051505	020123	
7991	051300	042522	027107	044440	
7992	051306	041516	051117	042522	
7993	051314	052103	000		
7994	051317	115	030522	044440	EM4016: .ASCIZ /MR1 INCORRECT/
7995	051324	041516	051117	042522	
7996	051332	052103	000		
7997	051335	105	041503	050040	EM4017: .ASCIZ /ECC PAT REG INCORRECT/
7998	051342	052101	051040	043505	
7999	051350	044440	041516	051117	
8000	051356	042522	052103	000	
8001	051363	115	030522	044440	EMW2: .ASCIZ /MR1 INCORRECT ON 1ST DOWNWARD TRANSITION OF MAINT CLOCK/
8002	051370	041516	051117	042522	
8003	051376	052103	047440	020116	

8004	051404	051461	020124	047504
8005	051412	047127	040527	042122
8006	051420	052040	040522	051516
8007	051426	051511	047511	020116
8008	051434	043117	046440	044501
8009	051442	052116	041440	047514
8010	051450	045503	000	
8011	051453	115	030522	044440
8012	051460	041516	051117	042522
8013	051466	052103	047440	020116
8014	051474	047062	020104	047504
8015	051502	047127	040527	042122
8016	051510	052040	040522	051516
8017	051516	052111	047511	020116
8018	051524	043117	046440	044501
8019	051532	052116	041440	047514
8020	051540	045503	000	

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

.SBTTL DATA BUFFERS

.EVEN  
NPRBUF :

8021					
8022					
8023		051544			
8024	051544				
8025	051544	000	000	.BYTE	0,0
8026	051546	001	001	.BYTE	1,1
8027	051550	002	002	.BYTE	2,2
8028	051552	003	003	.BYTE	3,3
8029	051554	004	004	.BYTE	4,4
8030	051556	005	005	.BYTE	5,5
8031	051560	006	006	.BYTE	6,6
8032	051562	007	007	.BYTE	7,7
8033	051564	010	010	.BYTE	10,10
8034	051566	011	011	.BYTE	11,11
8035	051570	012	012	.BYTE	12,12
8036	051572	013	013	.BYTE	13,13
8037	051574	014	014	.BYTE	14,14
8038	051576	015	015	.BYTE	15,15
8039	051600	016	016	.BYTE	16,16
8040	051602	017	017	.BYTE	17,17
8041	051604	020	020	.BYTE	20,20
8042	051606	021	021	.BYTE	21,21
8043	051610	022	022	.BYTE	22,22
8044	051612	023	023	.BYTE	23,23
8045	051614	024	024	.BYTE	24,24
8046	051616	025	025	.BYTE	25,25
8047	051620	026	026	.BYTE	26,26
8048	051622	027	027	.BYTE	27,27
8049	051624	030	030	.BYTE	30,30
8050	051626	031	031	.BYTE	31,31
8051	051630	032	032	.BYTE	32,32
8052	051632	033	033	.BYTE	33,33
8053	051634	034	034	.BYTE	34,34
8054	051636	035	035	.BYTE	35,35
8055	051640	036	036	.BYTE	36,36
8056	051642	037	037	.BYTE	37,37
8057	051644	040	040	.BYTE	40,40
8058	051646	041	041	.BYTE	41,41
8059	051650	042	042	.BYTE	42,42

8060	051652	043	043	.BYTE	43,43
8061	051654	044	044	.BYTE	44,44
8062	051656	045	045	.BYTE	45,45
8063	051660	046	046	.BYTE	46,46
8064	051662	047	047	.BYTE	47,47
8065	051664	050	050	.BYTE	50,50
8066	051666	051	051	.BYTE	51,51
8067	051670	052	052	.BYTE	52,52
8068	051672	053	053	.BYTE	53,53
8069	051674	054	054	.BYTE	54,54
8070	051676	055	055	.BYTE	55,55
8071	051700	056	056	.BYTE	56,56
8072	051702	057	057	.BYTE	57,57
8073	051704	060	060	.BYTE	60,60
8074	051706	061	061	.BYTE	61,61
8075	051710	062	062	.BYTE	62,62
8076	051712	063	063	.BYTE	63,63
8077	051714	064	064	.BYTE	64,64
8078	051716	065	065	.BYTE	65,65
8079	051720	066	066	.BYTE	66,66
8080	051722	067	067	.BYTE	67,67
8081	051724	070	070	.BYTE	70,70
8082	051726	071	071	.BYTE	71,71
8083	051730	072	072	.BYTE	72,72
8084	051732	073	073	.BYTE	73,73
8085	051734	074	074	.BYTE	74,74
8086	051736	075	075	.BYTE	75,75
8087	051740	076	076	.BYTE	76,76
8088	051742	077	077	.BYTE	77,77
8089	051744	100	100	.BYTE	100,100
8090	051746	101	101	.BYTE	101,101
8091	051750	102	102	.BYTE	102,102
8092	051752	103	103	.BYTE	103,103
8093	051754	000000		HEAD1: .WORD	000000
8094	051756	140000		.WORD	140000
8095	051760	140000		.WORD	140000
8096	051762	000000		HEAD2: .WORD	000000
8097	051764	040000		.WORD	040000
8098	051766	040000		.WORD	040000
8099	051770	000000		HEAD3: .WORD	000000
8100	051772	100000		.WORD	100000
8101	051774	100000		.WORD	100000
8102	051776	000300		HEAD4: .WORD	000300
8103	052000	040057		.WORD	040057
8104	052002	040356		.WORD	040356
8105	052004	000100		HEAD5: .WORD	000100
8106	052006	040000		.WORD	040000
8107	052010	040100		.WORD	040100
8108	052012	000100		.WORD	000100
8109	052014	140001		.WORD	140001
8110	052016	140101		.WORD	140101
8111	052020	000200		HEAD6: .WORD	000200
8112	052022	140000		.WORD	140000
8113	052024	140000		.WORD	140000
8114	052026	000200		.WORD	000200
8115	052030	140001		.WORD	140001

8116	052032	140201				.WORD	140201
8117	052034	000400			HEAD7:	.WORD	000400
8118	052036	140000				.WORD	140000
8119	052040	140000				.WORD	140000
8120	052042	000400				.WORD	000400
8121	052044	040001				.WORD	040001
8122	052046	040401				.WORD	040401
8123	052050	000140			HEAD8:	.WORD	000140
8124	052052	040000				.WORD	040000
8125	052054	040140				.WORD	040140
8126	052056	000140				.WORD	000140
8127	052060	140001				.WORD	140001
8128	052062	140101				.WORD	140101
8129	052064	000300			HEAD10:	.WORD	000300
8130	052066	140000				.WORD	140000
8131	052070	140000				.WORD	140000
8132	052072	000000			HEAD11:	.WORD	000000
8133	052074	141000				.WORD	141000
8134	052076	141000				.WORD	141000
8135	052100	000000	000000	000000	ECC1:	.WORD	0,0,0,0,0,0,0,0
8136	052106	000000	000000	000000			
8137	052114	000000	000000				
8138	052120	005001			ECC2:	.WORD	005001
8139	052122	040040				.WORD	040040
8140	052124	020004				.WORD	020004
8141	052126	000064				.WORD	000064
8142	052130	000000				.WORD	000000
8143	052132	000000				.WORD	000000
8144	052134	000000				.WORD	000000
8145	052136	000000				.WORD	000000
8146	052140	177777			ECC3:	.WORD	177777
8147	052142	177777				.WORD	177777
8148	052144	177777				.WORD	177777
8149	052146	177777				.WORD	177777
8150	052150	177777				.WORD	177777
8151	052152	177777				.WORD	177777
8152	052154	177777				.WORD	177777
8153	052156	177777				.WORD	177777
8154	052160	177777				.WORD	177777
8155	052162	177777				.WORD	177777
8156	052164	001252			OP11:	.WORD	001252
8157	052166	141123				.WORD	141123
8158	052170	140371				.WORD	140371
8159	052172	001251				.WORD	001251
8160	052174	141123				.WORD	141123
8161	052176	140372				.WORD	140372
8162	052200	001257				.WORD	001257
8163	052202	141123				.WORD	141123
8164	052204	140374				.WORD	140374
8165	052206	001243				.WORD	001243
8166	052210	141123				.WORD	141123
8167	052212	140360				.WORD	140360
8168	052214	001273				.WORD	001273
8169	052216	141123				.WORD	141123
8170	052220	140350				.WORD	140350
8171	052222	001213				.WORD	001213

8172	052224	141123	.WORD	141123
8173	052226	140330	.WORD	140330
8174	052230	001353	.WORD	001353
8175	052232	141123	.WORD	141123
8176	052234	140270	.WORD	140270
8177	052236	001053	.WORD	001053
8178	052240	141123	.WORD	141123
8179	052242	140170	.WORD	140170
8180	052244	001653	.WORD	001653
8181	052246	141123	.WORD	141123
8182	052250	140770	.WORD	140770
8183	052252	000253	.WORD	000253
8184	052254	141123	.WORD	141123
8185	052256	141370	.WORD	141370
8186	052260	003253	.WORD	003253
8187	052262	141123	.WORD	141123
8188	052264	142370	.WORD	142370
8189	052266	005253	.WORD	005253
8190	052270	141123	.WORD	141123
8191	052272	144370	.WORD	144370
8192	052274	011253	.WORD	011253
8193	052276	141123	.WORD	141123
8194	052300	150370	.WORD	150370
8195	052302	021253	.WORD	021253
8196	052304	141123	.WORD	141123
8197	052306	160370	.WORD	160370
8198	052310	041253	.WORD	041253
8199	052312	141123	.WORD	141123
8200	052314	100370	.WORD	100370
8201	052316	101253	.WORD	101253
8202	052320	141123	.WORD	141123
8203	052322	040370	.WORD	040370
8204	052324	001253	.WORD	001253
8205	052326	141122	.WORD	141122
8206	052330	140371	.WORD	140371
8207	052332	001253	.WORD	001253
8208	052334	141121	.WORD	141121
8209	052336	140372	.WORD	140372
8210	052340	001253	.WORD	001253
8211	052342	141127	.WORD	141127
8212	052344	140374	.WORD	140374
8213	052346	001253	.WORD	001253
8214	052350	141133	.WORD	141133
8215	052352	140360	.WORD	140360
8216	052354	001253	.WORD	001253
8217	052356	141103	.WORD	141103
8218	052360	140350	.WORD	140350
8219	052362	001253	.WORD	001253
8220	052364	141163	.WORD	141163
8221	052366	140330	.WORD	140330
8222	052370	001253	.WORD	001253
8223	052372	141023	.WORD	141023
8224	052374	140270	.WORD	140270
8225	052376	001253	.WORD	001253
8226	052400	141323	.WORD	141323
8227	052402	140170	.WORD	140170

8228	052404	001253	.WORD	001253
8229	052406	141523	.WORD	141523
8230	052410	140770	.WORD	140770
8231	052412	001253	.WORD	001253
8232	052414	140123	.WORD	140123
8233	052416	141370	.WORD	141370
8234	052420	001253	.WORD	001253
8235	052422	143123	.WORD	143123
8236	052424	142370	.WORD	142370
8237	052426	001253	.WORD	001253
8238	052430	145123	.WORD	145123
8239	052432	144370	.WORD	144370
8240	052434	001253	.WORD	001253
8241	052436	151123	.WORD	151123
8242	052440	150370	.WORD	150370
8243	052442	001253	.WORD	001253
8244	052444	161123	.WORD	161123
8245	052446	160370	.WORD	160370
8246	052450	001250	.WORD	001250
8247	052452	141123	.WORD	141123
8248	052454	140373	.WORD	140373
8249	052456	141253	.WORD	141253
8250	052460	141123	.WORD	141123
8251	052462	000370	.WORD	000370
8252	052464	000240	.WORD	000240
8253	052466	140000	.WORD	140000
8254	052470	140240	.WORD	140240
8255	052472	000240	.WORD	000240
8256	052474	140000	.WORD	140000
8257	052476	140240	.WORD	140240
8258	052500	000240	.WORD	000240
8259	052502	140000	.WORD	140000
8260	052504	140240	.WORD	140240
8261	052506	000240	.WORD	000240
8262	052510	140000	.WORD	140000
8263	052512	140240	.WORD	140240
8264	052514	000240	.WORD	000240
8265	052516	140000	.WORD	140000
8266	052520	140240	.WORD	140240
8267	052522	000240	.WORD	000240
8268	052524	140000	.WORD	140000
8269	052526	140240	.WORD	140240
8270	052530	000240	.WORD	000240
8271	052532	140000	.WORD	140000
8272	052534	140240	.WORD	140240
8273	052536	000240	.WORD	000240
8274	052540	140000	.WORD	140000
8275	052542	140240	.WORD	140240
8276	052544	000240	.WORD	000240
8277	052546	140000	.WORD	140000
8278	052550	140240	.WORD	140240
8279	052552	000240	.WORD	000240
8280	052554	140000	.WORD	140000
8281	052556	140240	.WORD	140240
8282	052560	000240	.WORD	000240
8283	052562	140000	.WORD	140000

OP12:

8284	052564	140240	.WORD	140240
8285	052566	000240	.WORD	000240
8286	052570	140000	.WORD	140000
8287	052572	140240	.WORD	140240
8288	052574	000240	.WORD	000240
8289	052576	140000	.WORD	140000
8290	052600	140240	.WORD	140240
8291	052602	000240	.WORD	000240
8292	052604	140000	.WORD	140000
8293	052606	140240	.WORD	140240
8294	052610	000240	.WORD	000240
8295	052612	140000	.WORD	140000
8296	052614	140240	.WORD	140240
8297	052616	000240	.WORD	000240
8298	052620	140000	.WORD	140000
8299	052622	140240	.WORD	140240
8300	052624	000240	.WORD	000240
8301	052626	140000	.WORD	140000
8302	052630	140240	.WORD	140240
8303	052632	000240	.WORD	000240
8304	052634	140000	.WORD	140000
8305	052636	140240	.WORD	140240
8306	052640	000240	.WORD	000240
8307	052642	140000	.WORD	140000
8308	052644	140240	.WORD	140240
8309	052646	000240	.WORD	000240
8310	052650	140000	.WORD	140000
8311	052652	140240	.WORD	140240
8312	052654	000240	.WORD	000240
8313	052656	140000	.WORD	140000
8314	052660	140240	.WORD	140240
8315	052662	000240	.WORD	000240
8316	052664	140000	.WORD	140000
8317	052666	140240	.WORD	140240
8318	052670	000240	.WORD	000240
8319	052672	140000	.WORD	140000
8320	052674	140240	.WORD	140240
8321	052676	000240	.WORD	000240
8322	052700	140000	.WORD	140000
8323	052702	140240	.WORD	140240
8324	052704	000240	.WORD	000240
8325	052706	140000	.WORD	140000
8326	052710	140240	.WORD	140240
8327	052712	000240	.WORD	000240
8328	052714	140000	.WORD	140000
8329	052716	140240	.WORD	140240
8330	052720	000240	.WORD	000240
8331	052722	140000	.WORD	140000
8332	052724	140240	.WORD	140240
8333	052726	000240	.WORD	000240
8334	052730	140000	.WORD	140000
8335	052732	140240	.WORD	140240
8336	052734	000240	.WORD	000240
8337	052736	140000	.WORD	140000
8338	052740	140240	.WORD	140240
8339	052742	000240	.WORD	000240



8340	052744	140000	.WORD	140000
8341	052746	140240	.WORD	140240
8342	052750	000240	.WORD	000240
8343	052752	140000	.WORD	140000
8344	052754	140040	.WORD	140040
8345	052756	000240	.WORD	000240
8346	052760	140000	.WORD	140000
8347	052762	140040	.WORD	140040
8348	052764	000300	OP13: .WORD	000300
8349	052766	140000	.WORD	140000
8350	052770	140300	.WORD	140300
8351	052772	000300	.WORD	000300
8352	052774	140000	.WORD	140000
8353	052776	140300	.WORD	140300
8354	053000	000300	.WORD	000300
8355	053002	140000	.WORD	140000
8356	053004	140300	.WORD	140300
8357	053006	000300	.WORD	000300
8358	053010	140000	.WORD	140000
8359	053012	140300	.WORD	140300
8360	053014	000300	.WORD	000300
8361	053016	140000	.WORD	140000
8362	053020	140300	.WORD	140300
8363	053022	000300	.WORD	000300
8364	053024	140000	.WORD	140000
8365	053026	140300	.WORD	140300
8366	053030	000300	.WORD	000300
8367	053032	140000	.WORD	140000
8368	053034	140300	.WORD	140300
8369	053036	000300	.WORD	000300
8370	053040	140000	.WORD	140000
8371	053042	140300	.WORD	140300
8372	053044	000300	.WORD	000300
8373	053046	140000	.WORD	140000
8374	053050	140300	.WORD	140300
8375	053052	000300	.WORD	000300
8376	053054	140000	.WORD	140000
8377	053056	140300	.WORD	140300
8378	053060	000300	.WORD	000300
8379	053062	140000	.WORD	140000
8380	053064	140300	.WORD	140300
8381	053066	000300	.WORD	000300
8382	053070	140000	.WORD	140000
8383	053072	140300	.WORD	140300
8384	053074	000300	.WORD	000300
8385	053076	140000	.WORD	140000
8386	053100	140300	.WORD	140300
8387	053102	000300	.WORD	000300
8388	053104	140000	.WORD	140000
8389	053106	140300	.WORD	140300
8390	053110	000300	.WORD	000300
8391	053112	140000	.WORD	140000
8392	053114	140300	.WORD	140300
8393	053116	000300	.WORD	000300
8394	053120	140000	.WORD	140000
8395	053122	140300	.WORD	140300

8396	053124	000300	.WORD	000300
8397	053126	140000	.WORD	140000
8398	053130	140300	.WORD	140300
8399	053132	000300	.WORD	000300
8400	053134	140000	.WORD	140000
8401	053136	140300	.WORD	140300
8402	053140	000300	.WORD	000300
8403	053142	140000	.WORD	140000
8404	053144	140300	.WORD	140300
8405	053146	000300	.WORD	000300
8406	053150	140000	.WORD	140000
8407	053152	140300	.WORD	140300
8408	053154	000300	.WORD	000300
8409	053156	140000	.WORD	140000
8410	053160	140300	.WORD	140300
8411	053162	000300	.WORD	000300
8412	053164	140000	.WORD	140000
8413	053166	140300	.WORD	140300
8414	053170	000300	.WORD	000300
8415	053172	140000	.WORD	140000
8416	053174	140300	.WORD	140300
8417	053176	000300	.WORD	000300
8418	053200	140000	.WORD	140000
8419	053202	140300	.WORD	140300
8420	053204	000300	.WORD	000300
8421	053206	140000	.WORD	140000
8422	053210	140300	.WORD	140300
8423	053212	000300	.WORD	000300
8424	053214	140000	.WORD	140000
8425	053216	140300	.WORD	140300
8426	053220	000300	.WORD	000300
8427	053222	140000	.WORD	140000
8428	053224	140300	.WORD	140300
8429	053226	000300	.WORD	000300
8430	053230	140000	.WORD	140000
8431	053232	140300	.WORD	140300
8432	053234	000300	.WORD	000300
8433	053236	140000	.WORD	140000
8434	053240	140300	.WORD	140300
8435	053242	000300	.WORD	000300
8436	053244	140000	.WORD	140000
8437	053246	140300	.WORD	140300
8438	053250	000300	.WORD	000300
8439	053252	140000	.WORD	140000
8440	053254	140200	.WORD	140200
8441	053256	000300	.WORD	000300
8442	053260	140000	.WORD	140000
8443	053262	140300	.WORD	140300
8444	053264	000040	.WORD	000040
8445	053266	140000	.WORD	140000
8446	053270	140000	.WORD	140000
8447	053272	000040	.WORD	000040
8448	053274	140000	.WORD	140000
8449	053276	140040	.WORD	140040
8450	053300	000040	.WORD	000040
8451	053302	140000	.WORD	140000

OP14:

8452	053304	140040	.WORD	140040
8453	053306	000040	.WORD	000040
8454	053310	140000	.WORD	140000
8455	053312	140040	.WORD	140040
8456	053314	000040	.WORD	000040
8457	053316	140000	.WORD	140000
8458	053320	140040	.WORD	140040
8459	053322	000040	.WORD	000040
8460	053324	140000	.WORD	140000
8461	053326	140040	.WORD	140040
8462	053330	000040	.WORD	000040
8463	053332	140000	.WORD	140000
8464	053334	140040	.WORD	140040
8465	053336	000040	.WORD	000040
8466	053340	140000	.WORD	140000
8467	053342	140040	.WORD	140040
8468	053344	000040	.WORD	000040
8469	053346	140000	.WORD	140000
8470	053350	140040	.WORD	140040
8471	053352	000040	.WORD	000040
8472	053354	140000	.WORD	140000
8473	053356	140040	.WORD	140040
8474	053360	000040	.WORD	000040
8475	053362	140000	.WORD	140000
8476	053364	140040	.WORD	140040
8477	053366	000040	.WORD	000040
8478	053370	140000	.WORD	140000
8479	053372	140040	.WORD	140040
8480	053374	000040	.WORD	000040
8481	053376	140000	.WORD	140000
8482	053400	140040	.WORD	140040
8483	053402	000040	.WORD	000040
8484	053404	140000	.WORD	140000
8485	053406	140040	.WORD	140040
8486	053410	000040	.WORD	000040
8487	053412	140000	.WORD	140000
8488	053414	140040	.WORD	140040
8489	053416	000040	.WORD	000040
8490	053420	140000	.WORD	140000
8491	053422	140040	.WORD	140040
8492	053424	000040	.WORD	000040
8493	053426	140000	.WORD	140000
8494	053430	140040	.WORD	140040
8495	053432	000040	.WORD	000040
8496	053434	140000	.WORD	140000
8497	053436	140040	.WORD	140040
8498	053440	000040	.WORD	000040
8499	053442	140000	.WORD	140000
8500	053444	140040	.WORD	140040
8501	053446	000040	.WORD	000040
8502	053450	140000	.WORD	140000
8503	053452	140040	.WORD	140040
8504	053454	000040	.WORD	000040
8505	053456	140000	.WORD	140000
8506	053460	140040	.WORD	140040
8507	053462	000040	.WORD	000040

8508	053464	140000			.WORD	140000
8509	053466	140040			.WORD	140040
8510	053470	000040			.WORD	000040
8511	053472	140000			.WORD	140000
8512	053474	140040			.WORD	140040
8513	053476	000040			.WORD	000040
8514	053500	140000			.WORD	140000
8515	053502	140040			.WORD	140040
8516	053504	000040			.WORD	000040
8517	053506	140000			.WORD	140000
8518	053510	140040			.WORD	140040
8519	053512	000040			.WORD	000040
8520	053514	140000			.WORD	140000
8521	053516	140040			.WORD	140040
8522	053520	000040			.WORD	000040
8523	053522	140000			.WORD	140000
8524	053524	140040			.WORD	140040
8525	053526	000040			.WORD	000040
8526	053530	140000			.WORD	140000
8527	053532	140040			.WORD	140040
8528	053534	000040			.WORD	000040
8529	053536	140000			.WORD	140000
8530	053540	140040			.WORD	140040
8531	053542	000040			.WORD	000040
8532	053544	140000			.WORD	140000
8533	053546	140040			.WORD	140040
8534	053550	000040			.WORD	000040
8535	053552	140000			.WORD	140000
8536	053554	140040			.WORD	140040
8537	053556	000040			.WORD	000040
8538	053560	140000			.WORD	140000
8539	053562	140040			.WORD	140040
8540	053564	001253			.WORD	001253
8541	053566	141123			.WORD	141123
8542	053570	140370			.WORD	140370
8543	053572	177777	177777	177777	MOD2BF: .WORD	177777,177777,177777,177777,177777,177777
8544	053600	177777	177777	177777		
8545	053606	000400			BUFF: .BLKW	400
8546	054606	125252	125252	125252	ECCBUF: .WORD	125252,125252,125252,125252,125252,125252
8547	054614	125252	125252	125252		
8548	054622	125252	125252	177777	.WORD	125252,125252,177777,177777,177777,177777
8549	054630	177777	177777	177777		
8550	054636	177777	177777	177777	.WORD	177777,177777,177777,177777,000000,000000
8551	054644	177777	000000	000000		
8552	054652	000000	000000	000000	.WORD	000000,000000,000000,000000,000000,000000
8553	054660	000000	000000	000000		
8554	054666	052525	052525	052525	.WORD	052525,052525,052525,052525,052525,052525
8555	054674	025252	052525	052525		
8556	054702	052525	052525	121105	.WORD	052525,052525,121105,150442,064221,132110
8557	054710	150442	064221	132110		
8558	054716	055044	026422	013211	.WORD	055044,026422,013211,105504,042642,021321
8559	054724	105504	042642	021321		
8560	054732	110550	044264	022132	.WORD	110550,044264,022132,011055,104426,042213
8561	054740	011055	104426	042213		
8562	054746	133333	066666	155555	.WORD	133333,066666,155555,155555,133333,066666
8563	054754	155555	133333	066666		

8564	054762	066666	155555	155555	.WORD	066666,155555,155555,133333,133333,133333
8565	054770	133333	133333	133333		
8566	054776	133333	133333	133333	.WORD	133333,133333,133333,133333,177777,177777
8567	055004	133333	177777	177777		
8568	055012	177777	052525	052525	.WORD	177777,052525,052525,052525,177777,177777
8569	055020	052525	177777	177777		
8570	055026	052525	052525	177777	.WORD	052525,052525,177777,052525,177252,177252
8571	055034	052525	177252	177252		
8572	055042	172765	172765	072307	.WORD	172765,172765,072307,135143,156461,167230
8573	055050	135143	156461	167230		
8574	055056	073514	035646	016723	.WORD	073514,035646,016723,107351,143564,061672
8575	055064	107351	143564	061672		
8576	055072	030735	114356	046167	.WORD	030735,114356,046167,123073,151453,164616
8577	055100	123073	151453	164616		
8578	055106	125252	125252	125252	.WORD	125252,125252,125252,125252,125252,125252
8579	055114	125252	125252	125252		
8580	055122	125252	125252	177777	.WORD	125252,125252,177777,177777,177777,177777
8581	055130	177777	177777	177777		
8582	055136	177777	177777	177777	.WORD	177777,177777,177777,177777,000000,000000
8583	055144	177777	000000	000000		
8584	055152	000000	000000	000000	.WORD	000000,000000,000000,000000,000000,000000
8585	055160	000000	000000	000000		
8586	055166	052525	052525	052525	.WORD	052525,052525,052525,025252,052525,052525
8587	055174	025252	052525	052525		
8588	055202	052525	052525	121105	.WORD	052525,052525,121105,150442,064221,132110
8589	055210	150442	064221	132110		
8590	055216	055044	026422	013211	.WORD	055044,026422,013211,105504,042642,021321
8591	055224	105504	042642	021321		
8592	055232	110550	044264	022132	.WORD	110550,044264,022132,011055,104426,042213
8593	055240	011055	104426	042213		
8594	055246	133333	066666	155555	.WORD	133333,066666,155555,155555,133333,066666
8595	055254	155555	133333	066666		
8596	055262	066666	155555	155555	.WORD	066666,155555,155555,133333,133333,133333
8597	055270	133333	133333	133333		
8598	055276	133333	133333	133333	.WORD	133333,133333,133333,133333,177777,177777
8599	055304	133333	177777	177777		
8600	055312	177777	052525	052525	.WORD	177777,052525,052525,052525,177777,177777
8601	055320	052525	177777	177777		
8602	055326	052525	052525	177777	.WORD	052525,052525,177777,052525,177252,177252
8603	055334	052525	177252	177252		
8604	055342	172765	172765	072307	.WORD	172765,172765,072307,135143,156461,167230
8605	055350	135143	156461	167230		
8606	055356	073514	035646	016723	.WORD	073514,035646,016723,107351,143564,061672
8607	055364	107351	143564	061672		
8608	055372	030735	114356	046167	.WORD	030735,114356,046167,123073,151453,164616
8609	055400	123073	151453	164616		
8610	055406	125252	125252	125252	.WORD	125252,125252,125252,125252,125252,125252
8611	055414	125252	125252	125252		
8612	055422	125252	125252	177777	.WORD	125252,125252,177777,177777,177777,177777
8613	055430	177777	177777	177777		
8614	055436	177777	177777	177777	.WORD	177777,177777,177777,177777,000000,000000
8615	055444	177777	000000	000000		
8616	055452	000000	000000	000000	.WORD	000000,000000,000000,000000,000000,000000
8617	055460	000000	000000	000000		
8618	055466	052525	052525	052525	.WORD	052525,052525,052525,025252,052525,052525
8619	055474	025252	052525	052525		

CZR6DCO RK611 DSKLS PRT4  
CZR6DC.P11 10-JUN-80 15:32

MACY11 30A(1052) 10-JUN-80 15:34 PAGE 158  
DATA BUFFERS

B 13

SEQ 0157

8620	055502	052525	052525	121105	.WORD	052525,052525,121105,150442,064221,132110
8621	055510	150442	064221	132110		
8622	055516	055044	026422	013211	.WORD	055044,026422,013211,105504,042642,021321
8623	055524	105504	042642	021321		
8624	055532	110550	044264	022132	.WORD	110550,044264,022132,011055,104426,042213
8625	055540	011055	104426	042213		
8626	055546	177777	177777	177777	.WORD	177777,177777,177777,177777,177777,177777
8627	055554	177777	177777	177777		
8628	055562	177777	177777	177777	.WORD	177777,177777,177777,177777,177777,177777
8629	055570	177777	177777	177777		
8630	055576	177777	177777	177777	.WORD	177777,177777,177777,177777
8631	055604	177777				
8632	055606	177777	177777	177777	.WORD	177777,177777,177777,177777,177777,177777
8633	055614	177777	177777	177777		
8634	000001				.END	

ABASE = 177440	1002#	1237	1278	
ABORT = 042344	6123	7349#		
ACDW1 = 000000	1237	1280		
ACDW2 = 000000	1237	1281		
ACLO = 000010	1097#			
ACPUOP = 000000	1237	1252		
ADDW0 = 000000	1237			
ADDW1 = 000000	1237			
ADDW10 = 000000	1237			
ADDW11 = 000000	1237			
ADDW12 = 000000	1237			
ADDW13 = 000000	1237			
ADDW14 = 000000	1237			
ADDW15 = 000000	1237			
ADDW2 = 000000	1237			
ADDW3 = 000000	1237			
ADDW4 = 000000	1237			
ADDW5 = 000000	1237			
ADDW6 = 000000	1237			
ADDW7 = 000000	1237			
ADDW8 = 000000	1237			
ADDW9 = 000000	1237			
ADEVCT = 000000	1237	1243		
ADEVN = 000000	1237	1279		
AENV = 000000	1237	1248		
AENVN = 000000	1237	1249		
AFATAL = 000000	1237	1240		
AMADR1 = 000000	1237	1265		
AMADR2 = 000000	1237	1269		
AMADR3 = 000000	1237	1272		
AMADR4 = 000000	1237	1275		
AMAMS1 = 000000	1237	1259		
AMAMS2 = 000000	1237	1267		
AMAMS3 = 000000	1237	1270		
AMAMS4 = 000000	1237	1273		
AMSGAD = 000000	1237	1245		
AMSGLG = 000000	1237	1246		
AMSGTY = 000000	1237	1239		
AMTYP1 = 000000	1237	1260		
AMTYP2 = 000000	1237	1268		
AMTYP3 = 000000	1237	1271		
AMTYP4 = 000000	1237	1274		
APASS = 000000	1237	1242		
APRIOR = 000240	1001#	1237		
APTCSU = 000040	6396#	6482		
APTENV = 000001	6352	6394#	6428	6475
APTSIZ = 000200	2131	6393#		
APTSPO = 000100	6354	6395#	6477	
ASWREG = 000000	1237	1250		
ATESTN = 000000	1237	1241		
AUNIT = 000000	1237	1244		
AUSWR = 000000	1237	1251		
AVECT1 = 000210	1000#	1237	1276	
AVECT2 = 000000	1237	1277		
BAI = 000020	1060#			
BA16 = 000400	1045#			







CZR6DCO RK611 DSKLS PRT4  
CZR6DC.P11 10-JUN-80 15:32

MACY11 30A(1052) 10-JUN-80 15:34 PAGE 163  
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0161

DH077B 044170  
DH134A 044214  
DH134B 044240  
DH144A 044266  
DH144B 044365  
DH151A 044462  
DI = 040000  
DISPLA 001142  
DISPRE 000174  
DLT = 100000  
DMD = 000040

7280	7516#													
7288	7520#													
7290	7524#													
7302	7528#													
7304	7539#													
7314	7550#													
1052#														
1210#	2119*	2127*	6000*	6415*										
1142#	2127													
1071#														
1113#	2231	2240	2241	2277	2286	2287	2324	2333	2334	2370	2379	2380		
2416	2425	2426	2463	2472	2473	2521	2528	2529	2578	2585	2586	2632		
2639	2640	2687	2694	2695	2737	2742	2743	2830	2836	2837	2840	2841		
2887	2921	2927	2928	2932	2933	2934	2937	2969	2998	3006	3007	3010		
3011	3079	3087	3088	3091	3092	3160	3168	3169	3172	3173	3239	3247		
3248	3251	3252	3321	3329	3330	3333	3334	3402	3410	3411	3419	3421		
3422	3516	3530	3531	3534	3535	3603	3611	3612	3615	3616	3693	3701		
3702	3710	3712	3713	3761	3796	3804	3805	3813	3815	3816	3864	3900		
3908	3909	3917	3919	3920	4004	4012	4013	4021	4023	4024	4109	4117		
4118	4126	4128	4129	4235	4243	4244	4252	4254	4255	4354	4362	4363		
4371	4373	4374	4458	4466	4467	4470	4471	4503	4504	4548	4556	4557		
4560	4561	4593	4594	4638	4646	4647	4650	4651	4683	4684	4728	4736		
4737	4740	4741	4773	4774	4821	4827	4828	4831	4832	4870	4933	4939		
4940	4943	4944	4982	5047	5053	5054	5057	5058	5096	5154	5160	5161		
5164	5165	5203	5286	5287	5347	5353	5354	5357	5358	5396	5483	5484		
5554	5560	5561	5565	5566	5608	5679	5685	5686	5690	5691	5733	6228		
6245	6246	6259	6260	6286	6287	6299	6300	6326	6329	6330	6331	6332		
6335	6336	6337	6338											
1094#														
1101#														
1099#														
1078#														
1058#														
1105#														
906#	1209	2118												
1087#														
1080#														
1302	7109#													
1307	7110#													
1313	1319	1325	1331	1337	1343	1349	1355	1361	1367	1373	1379	1385		
1391	1397	1403	1409	1415	1422	1429	1436	1443	1450	1457	1464	1471		
1478	1485	1492	1499	7113#										
1505	1511	1517	1523	1529	7116#									
1535	1541	1547	1614	1620	1626	1633	1640	1647	1875	1881	1887	1893		
1899	1905	7120#												
1553	7123#													
1559	1953	7125#												
1565	1608	7127#												
1571	1577	1583	1589	1595	1601	7129#								
1653	1659	1665	7131#											
1671	1677	1683	1695	1701	1707	1719	1725	1731	1743	1749	1755	1767		
1773	1779	1791	1797	1803	1815	1821	1827	1839	1845	1851	7135#			
1689	1713	1737	1761	1785	1809	1833	1857	7138#						
1863	1869	2007	7140#											
1911	1917	1923	1929	1935	1941	1947	1959	1965	1971	1977	1983	1989		
1995	7142#													

DRA = 000001  
DRDY = 000200  
DROT = 000040  
DRPAR = 000010  
DRVMSK = 000007  
DSC = 040000  
DSWR = 177570  
DTE = 010000  
DTYE = 000040  
DT000 041046  
DT002 041052  
DT003 041074  
  
DT041 041114  
DT046 041144  
  
DT051 041164  
DT052 041176  
DT053 041212  
DT054 041222  
DT071 041236  
DT074 041264  
  
DT077 041306  
DT134 041320  
DT144 041332

DT151	041372	2001	7148#												
ECCBUF	054606	5155	5217	5302	5348	5410	5439	5499	5680	5745	8546#				
ECCGEN	034526	5011	5125	5231	5424	5638	5763	5797	6198#						
ECCHI	003266	2063#	4993*	5107*	5212*	5249	5405*	5446	5617*	5742*	5817	6200	6206	6211*	
		6219	6220*	6221*	6222										
ECCL0	003270	2064#	4994*	5108*	5213*	5406*	5618*	5743*	6210*	6215	6216*	6217*			
ECCW =	020000	1121#	2887	2969	3419	3710	3761	3813	3864	3917	4021	4126	4252	4371	
		4870	4982	5096	5203	5241	5268	5396	5434	5465	5608	5733	5804	5836	
ECCXOR	003272	2065#	6202*	6208*	6212										
ECC1	052100	4822	4882	8135#											
ECC2	052120	4934	4995	8138#											
ECC3	052140	5048	5109	9146#											
ECH =	000100	1081#													
EMTVEC =	000030	995#	2102*	2103*											
EMW	001310	1305#	4864*	4880*	4976*	4992*	5090*	5106*	5200*	5214*	5246*	5273*	5393*	5407*	
		5443*	5470*	5602*	5623*	5649*	5727*	5748*	5774*	5781*	5812*	5815*	5842*	6251*	
		6265*	6292*	6305*											
EMW2	051363	6251	6292	8001#											
EMW4	051453	6265	6305	8011#											
EM000	044501	1300	7555#												
EM300	044546	1311	1317	1323	7562#										
EM301	044623	1329	1335	1341	7570#										
EM302	044701	1347	1353	1359	7578#										
EM303	044760	1365	1371	1377	7586#										
EM304	045036	1383	1389	1395	7594#										
EM305	045115	1401	1407	1413	7602#										
EM306	045175	1420	1427	1434	7611#										
EM307	045277	1441	1448	1455	7623#										
EM308	045376	1462	1469	1476	7635#										
EM309	045476	1483	1490	1497	7647#										
EM310	045576	1503	1509	1515	7659#										
EM311	045663	1557	1563	7668#											
EM312	045732	1569	1575	7675#											
EM313	045777	1581	1587	7682#											
EM314	046043	1593	1599	7689#											
EM315	046112	1606	7696#												
EM316	046206	1612	1618	1624	7707#										
EM317	046253	1631	1638	1645	7714#										
EM318	046351	1651	1657	1663	7726#										
EM319	046410	1669	1675	1681	1687	7732#									
EM320	046461	1693	1699	1705	1711	7739#									
EM321	046545	1717	1723	1729	1735	7748#									
EM322	046631	1741	1747	1753	1759	7757#									
EM323	046714	1765	1771	1777	1783	7766#									
EM324	047007	1789	1795	1801	1807	7776#									
EM325	047102	1813	1819	1825	1831	7786#									
EM326	047172	1837	1843	1849	1855	7796#									
EM327	047260	4864	4976	5090	5200	5393	5602	5727	7805#						
EM328	047332	4880	4992	5106	5214	5407	7812#								
EM329	047411	1861	7820#												
EM330	047462	1867	7827#												
EM331	047525	1873	1879	1885	7833#										
EM332	047612	1891	1897	1903	7842#										
EM333	047677	5246	5443	5815	7851#										
EM334	047735	5273	5470	5842	7857#										
EM335	047773	1909	1915	1921	1927	1933	1939	1945	7863#						

EM336	050047	1951	7871#											
EM337	050124	1957	1963	1969	1975	1981	1987	1993	7879#					
EM338	050203	5623	5748	5781	7887#									
EM339	050271	5649	5774	5812	7897#									
EM340	050375	1999	2005	7909#										
EM4000	050442	1312	1330	1348	1366	1384	1402	1421	1442	1463	1484	1504	1558	1613
		1632	1652	1670	1694	1718	1742	1766	1790	1814	1838	1874	1892	1910
		1952	1958	7916#										
EM4001	050460	1318	1336	1354	1372	1390	1408	7919#						
EM4002	050501	1324	1342	1360	1378	1396	1414	7922#						
EM4003	050522	1428	1449	1470	1491	7925#								
EM4004	050552	1435	1456	1477	1498	7929#								
EM4005	050602	1510	1619	1639	1658	1676	1700	1724	1748	1772	1796	1820	1844	1880
		1898	1916	1964	7933#									
EM4006	050620	1516	1625	1646	1664	1682	1706	1730	1754	1778	1802	1826	1850	1886
		1904	1922	1970	7936#									
EM4007	050644	1522	1928	1976	7940#									
EM4008	050672	1528	1934	1982	7944#									
EM4009	050717	1534	7948#											
EM4010	050767	1540	7955#											
EM4011	051037	1546	7962#											
EM4012	051115	1552	7970#											
EM4013	051155	1564	1688	1712	1736	1760	1784	1808	1832	1856	7976#			
EM4014	051223	1570	1582	1594	1946	1994	7983#							
EM4015	051257	1576	1588	1600	1940	1988	7988#							
EM4016	051317	1607	7994#											
EM4017	051335	1862	1868	2000	2006	7997#								
ERRCNT	003250	2056#	2135*	6118*	6121									
ERRVEC=	000004	988#	2116	2117*	2128*	5951	5952*	5954*	5957*					
E.ASOF	003216	2038#												
E.BA	003204	2033#	2754*	2765	5302*	5313	5499*	5510	7116	7145				
E.CS1	003200	2031#	2247*	2250	2293*	2296	2340*	2343	2386*	2389	2432*	2435	2479*	2482
		2514*	2526	2535	2571*	2583	2592	2625*	2637	2646	2680*	2692	2701	2714
		2716*	2751*	2756	2786	2842*	2849	2857	2873	3286*	3289	3368*	3371	3416*
		3457	3474*	3475*	3483*	3484	3504*	3569	3650*	3653	3707*	3747	3810*	3850
		3914*	3954	4018*	4058	4123*	4164	4181*	4182*	4190*	4191	4249*	4290	4307*
		4308*	4316*	4317	4368*	4409	4426*	4427*	4435*	4436	4510*	4513	4526*	4527
		4600*	4603	4616*	4617	4690*	4693	4706*	4707	4780*	4783	4796*	4797	5293*
		5304	5490*	5501	6181	7113	7116	7120	7125	7131	7135	7142		
E.CS2	003210	2035#	2752*	2762	2777*	2792	2804*	3287*	3292	3369*	3374	3417*	3460	3505*
		3572	3651*	3656	3708*	3750	3811*	3853	3915*	3957	4019*	4061	4124*	4167
		4250*	4293	4369*	4412	4511*	4516	4601*	4606	4691*	4696	4781*	4786	5294*
		5307	5491*	5504	7116	7120	7131	7135	7142					
E.DA	003206	2034#	2513*	2525	2570*	2582	2605*	2606	2624*	2636	2659*	2660	2679*	2691
		3044	3125	3206	5301*	5322	5498*	5519	6149*	6150*	6151	6153*	6154*	6155
		6157*	6172	6179	7129	7145								
E.DB	003222	2040#	2778*	2798	7123									
E.DCVL	003220	2039#	2512*	2524	2548*	2549	2551*	2552	2569*	2581	2623*	2635	2678*	2690
		3047	3128	3209	5300*	5319	5497*	5516	6148*	6158*	6164	7129	7145	
E.DS	003212	2036#												
E.ECPS	003232	2044#												
E.ECPT	003234	2045#	4881*	4898	5012	5126	5233	5426	5639	5764	5798	5854*	5856	6222*
		6223*	7140	7148										
E.ER	003214	2037#	2755*	2771	2795	3288*	3295	3370*	3377	3418*	3463	3473*	3506*	3575
		3652*	3659	3709*	3753	3812*	3856	3916*	3960	3968*	4020*	4064	4072*	4125*
		4170	4180*	4251*	4296	4306*	4370*	4415	4425*	4512*	4519	4602*	4609	4692*

E.MR1	003224	4699	4782*	4789	5295*	5310	5492*	5507	7116	7120	7131	7135	7142	
		2041#	2887*	2888	2969*	2970	3419*	3467	3710*	3757	3761*	3813*	3860	3864*
		3917*	3964	4021*	4068	4126*	4174	4252*	4300	4371*	4419	4870*	4987*	5096*
		5203*	5241*	5268*	5396*	5434*	5465*	5608*	5733*	5804*	5830*	6228*	6229*	6234*
E.MR2	003226	6239*	6244*	6248	6258*	6262	6280*	6285*	6289	6298*	6302	7110	7127	7138
		2042#	2248*	2256	2294*	2302	2341*	2349	2387*	2395	2433*	2441	2480*	2488
		2541	2598	2652	2707	6171*	7113							
E.MR3	003230	2043#	2249*	2259	2295*	2305	2342*	2352	2388*	2398	2434*	2444	2481*	2491
		2544	2601	2655	2710	6192*	7113							
E.SPAC	003236	2046#												
E.WC	003202	2032#	2753*	2768	5303*	5316	5500*	5513	7116	7145				
FMTE	= 000020	1079#												
GNS	= ***** U	1141	2155	5883	5890	7092	7093	7094	7095	7096	7098	7100	7101	7102
		7103	7104	7105	7106									
GO	= 000001	1042#	3474	4181	4307	4426	4510	4600	4690	4780	5293	5490		
GTSWR	= 104406	2150	7098#											
HDRCNT	003310	2071#	3420*	3423	3470*	3471	3711*	3760*	3814*	3863*	3918*	3967*	4022*	4071*
		4127*	4130	4177*	4178	4253*	4256	4303*	4304	4372*	4375	4422*	4423	7135
		7138												
HEADER	003302	2070#	3020	3101	3182	6135*	6136*	6137*	6138*	6139*	6140*	6141*	6142*	6143
		6144*	6145*	6146	6147*									
HEAD1	051754	2860	2946	4842	4954	5068	5174	5367	8093#					
HEAD10	052064	4570	4660	4750	8129#									
HEAD11	052072	5579	5704	8132#										
HEAD2	051762	3261	4480	8096#										
HEAD3	051770	3343	8099#											
HEAD4	051776	3625	8102#											
HEAD5	052004	3706	8105#											
HEAD6	052020	3809	8111#											
HEAD7	052034	3913	8117#											
HEAD8	052050	4017	8123#											
HT	= 000011	898#	6490	6531										
HVRC	= 000400	1083#	3506	3652	4072	4180	4306	4425	4602	4692	4782			
IDAE	= 002000	1085#												
IE	= 000100	1043#												
ILF	= 000001	1075#												
INCHDR	034220	2996	3077	3158	6135#									
INTR	= 000300	1038#												
IO1VEC	= 000020	993#	2100*	2101*										
IR	= 000100	1062#	2777	2804	3287	3369	3417	3505	3651	3708	3811	3915	4019	4124
		4250	4369	4511	4601	4691	4781	5294	5491					
LF	= 000012	899#	6525	6531										
MCLK	= 000400	1116#	2240	2286	2333	2379	2425	2472	2528	2585	2639	2694	2742	2836
		2927	2933	3006	3087	3168	3247	3329	3410	3530	3611	3701	3804	3908
		4012	4117	4243	4362	4466	4503	4556	4593	4646	4683	4736	4773	4827
		4939	5053	5160	5286	5353	5483	5560	5685	6245	6259	6286	6299	6326
		6329	6331	6335	6337									
MDS	= 001000	1065#												
MEMPAR	003274	2066#												
MERD	= 001000	1117#	6329	6337										
MESMSK	= 000017	1110#												
MEWD	= 002000	1118#	2887	2969	3419	3710	3761	3813	3864	3917	4021	4126	4252	4371
		4870	4982	5096	5203	5396	5608	5733	6228	6234	6258	6298		
MIND	= 000200	1115#	2932	2933	2934									
MOD2BF	053572	5555	5620	8543#										
MSP	= 000100	1114#	2840	3010	3091	3172	3251	3333	3421	3534	3615	3712	3815	3919

	4023	4128	4254	4373	4470	4560	4650	4740	4831	4943	5057	5164	5357
M1.BIT 003256	5565	5690											
	2059#	2844*	2864*	2881*	2939*	2950*	2963*	3013*	3024*	3037*	3094*	3105*	3118*
	3175*	3186*	3199*	3254*	3265*	3278*	3336*	3347*	3360*	3426*	3436*	3449*	3537*
	3548*	3561*	3618*	3629*	3642*	3715*	3725*	3739*	3818*	3828*	3842*	3922*	3932*
	3946*	4026*	4036*	4050*	4133*	4143*	4156*	4259*	4269*	4282*	4378*	4388*	4401*
	4473*	4484*	4497*	4563*	4574*	4587*	4653*	4664*	4677*	4743*	4754*	4767*	4834*
	4845*	4858*	4867*	4886	4887*	4946*	4957*	4970*	4979*	4999	5000*	5060*	5071*
	5084*	5093*	5113	5114*	5167*	5178*	5191*	5198*	5220	5221*	5252	5253*	5276
	5277*	5360*	5371*	5384*	5391*	5413	5414*	5449	5450*	5473	5474*	5569*	5583*
	5596*	5605*	5625	5626*	5694*	5708*	5721*	5730*	5750	5751*	5784	5785*	5821
	5822*	5845	5846*	6232	6278	6283	6324	7110					
M2.BIT 003260	2060#	4868*	4886*	4980*	4999*	5094*	5113*	5199*	5220*	5252*	5276*	5392*	5413*
	5449*	5473*	5606*	5625*	5731*	5750*	5784*	5821*	5845*	6237	6242	7110	
NED = 010000	1068#												
NEM = 004000	1067#												
NEWPAS 004300	2212#												
NPRBUF 051544	2739	2754	2774	8024#									
NXF = 000004	1077#												
OFFSET= 000015	1031#												
OFST = 000004	1096#												
OPI = 020000	1088#	3473	4180	4306	4425								
OP11 052164	3415	8156#											
OP12 052464	4122	8252#											
OP13 052764	4248	8348#											
OP14 053264	4367	8444#											
OPR001 042112	2162	7318#											
OPR002 042141	2165	2176	2192	7322#									
OPR003 042147	2172	7324#											
OPR004 042177	2183	7329#											
OPR005 042221	5918	7333#											
OPR006 042261	2159	7339#											
OR = 000200	1063#	2752	2777	3287	3369	3417	3505	3651	3708	3811	3915	4019	4124
	4250	4369	4511	4601									
PACK = 000003	1026#												
PARM 003316	1148	2076#											
PAT = 000020	1112#												
PCA = 004000	1119#	6229	6244	6280									
PCD = 010000	1120#	6229	6239	6285									
PGE = 002000	1066#												
PIP = 020000	1104#												
PIRQ = 177772	905#												
PIRQVE= 000240	999#												
PR.BIT 003254	2058#	2843*	2854*	2864	2867*	2870*	2881	2882*	2938*	2944*	2950	2953*	2956*
	2963	2964*	3012*	3018*	3024	3027*	3030*	3037	3038*	3093*	3099*	3105	3108*
	3111*	3118	3119*	3174*	3180*	3186	3189*	3192*	3199	3200*	3253*	3259*	3265
	3268*	3271*	3278	3279*	3335*	3341*	3347	3350*	3353*	3360	3361*	3425*	3431*
	3436	3439*	3442*	3449	3450*	3536*	3542*	3548	3551*	3554*	3561	3562*	3617*
	3623*	3629	3632*	3635*	3642	3643*	3714*	3720*	3725	3728*	3731*	3739	3740*
	3817*	3823*	3828	3831*	3834*	3842	3843*	3921*	3927*	3932	3935*	3938*	3946
	3947*	4025*	4031*	4036	4039*	4042*	4050	4051*	4132*	4138*	4143	4146*	4149*
	4156	4157*	4258*	4264*	4269	4272*	4275*	4282	4283*	4377*	4383*	4388	4391*
	4394*	4401	4402*	4472*	4478*	4484	4487*	4490*	4497	4498*	4562*	4568*	4574
	4577*	4580*	4587	4588*	4652*	4658*	4664	4667*	4670*	4677	4678*	4742*	4748*
	4754	4757*	4760*	4767	4768*	4833*	4839*	4845	4848*	4851*	4858	4859*	4866*
	4887	4888*	4945*	4951*	4957	4960*	4963*	4970	4971*	4978*	5000	5001*	5059*

5065*	5071	5074*	5077*	5084	5085*	5092*	5114	5115*	5166*	5172*	5178	5181*
5184*	5191	5192*	5197*	5221	5222*	5253	5254*	5277	5278*	5359*	5365*	5371
5374*	5377*	5384	5385*	5390*	5414	5415*	5450	5451*	5474	5475*	5568*	5576*
5583	5586*	5589*	5596	5597*	5604*	5626	5627*	5693*	5701*	5708	5711*	5714*
5721	5722*	5729*	5751	5752*	5785	5786*	5822	5823*	5846	5847*	6230	6322
7110												
922#	2212											
923#												
924#												
925#												
926#												
927#	1001	?050										
928#												
929#	2087	7037	7051									
902#	903											
903#												
994#	2106*	2107*	7036*	7037*	7050*	7051*						
2057#	4865*	4876*	4888	4891*	4894*	4977*	4988*	5001	5004*	5007*	5091*	5102*
5115	5118*	5121*	5196*	5209*	5222	5225*	5228*	5254	5257*	5260*	5278	5279*
5389*	5402*	5415	5418*	5421*	5451	5454*	5457*	5475	5476*	5603*	5614*	5627
5631*	5634*	5728*	5739*	5752	5756*	5759*	5786	5790*	5793*	5823	5826*	5829*
5847	5848*	6198	6235	6276	7110							
2847	2855	2871	2883	2941	2945	2957	2965	3015	3019	3031	3039	3096
3100	3112	3120	3177	3181	3193	3201	3256	3260	3272	3280	3338	3342
3354	3362	3428	3432	3443	3451	3539	3543	3555	3563	3620	3624	3636
3644	3717	3721	3732	3741	3820	3824	3835	3844	3924	3928	3939	3948
4028	4032	4043	4052	4135	4139	4150	4158	4261	4265	4276	4284	4380
4384	4395	4403	4475	4479	4491	4499	4565	4569	4581	4589	4655	4659
4671	4679	4745	4749	4761	4769	4836	4840	4852	4860	4948	4952	4964
4972	5062	5066	5078	5086	5169	5173	5185	5193	5362	5366	5378	5386
5572	5577	5590	5598	5697	5702	5715	5723	6322#				
6876	7101#											
1033#	2237	2247	2376	2386	2514	2571	2625	2680	4643	4690		
1123#	6229											
1035#												
6950	7102#											
2166	2177	2193	7103#									
1044#	3475	3483	4182	4190	4308	4316	4427	4435	4510	4600	4690	4780
5293	5490											
1030#												
6117	7105#											
1146	2079#											
989#												
1013#												
1008#	2233*	2279*	2326*	2372*	2418*	2465*	2523*	2580*	2634*	2689*	2739*	2749
2832*	2923*	3002*	3083*	3164*	3243*	3325*	3403*	3524*	3607*	3694*	3797*	3901*
4005*	4110*	4236*	4355*	4459*	4549*	4639*	4729*	4822*	4934*	5048*	5155*	5298
5348*	5495	5555*	5680*									
1006#	2230*	2237*	2244	2253*	2276*	2283*	2290	2299*	2323*	2330*	2337	2346*
2369*	2376*	2383	2392*	2415*	2422*	2429	2438*	2462*	2469*	2476	2485*	2520*
2526*	2532	2538*	2577*	2583*	2589	2595*	2631*	2637*	2643	2649*	2686*	2692*
2698	2704*	2736*	2740*	2746	2759*	2783	2789*	2829*	2833*	2848	2856	2872
2917*	2924*	2997*	3003*	3078*	3084*	3159*	3165*	3238*	3244*	3283	3320*	3326*
3365	3398*	3407*	3454	3479*	3480	3512*	3528*	3566	3602*	3608*	3647	3692*
3698*	3744	3795*	3801*	3847	3899*	3905*	3951	4003*	4009*	4055	4105*	4114*
4161	4186*	4187	4231*	4240*	4287	4312*	4313	4350*	4359*	4406	4431*	4432

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

RDBIT 035422

RDCHR = 104410  
 RDDATA= 000021  
 RDGATE= 100000  
 RDHEAD= 000025  
 RDLIN = 104411  
 RDOCT = 104412  
 RDY = 000200  
 RECAL = 000013  
 RESREG= 104414  
 RESTRT 003326  
 RESVEC= 000010  
 RKASOF= 000016  
 RKBA = 000004

RKCS1 = 000000









CZR6DCO RK611 DSKLS PRT4  
CZR6DC.P11 10-JUN-80 15:32

B 14  
MACY11 30A(1052) 10-JUN-80 15:34 PAGE 172  
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0170

T.CS2	003150	4523*	4527	4597*	4603	4613*	4617	4687*	4693	4703*	4707	4777*	4783	4793*
		4797	5290*	5304	5487*	5501	7113	7116	7120	7125	7131	7135	7142	
		2016#	2747*	2762	2784*	2792	2893*	3284*	3292	3366*	3374	3455*	3460	3481*
		3567*	3572	3648*	3656	3745*	3750	3848*	3853	3952*	3957	4056*	4061	4162*
		4167	4188*	4288*	4293	4314*	4407*	4412	4433*	4508*	4516	4524*	4598*	4606
		4614*	4688*	4696	4704*	4778*	4786	4794*	5291*	5307	5488*	5504	7116	7120
		7125	7131	7135	7142									
T.DA	003146	2015#	3042*	3044	3123*	3125	3204*	3206	5297*	5322	5494*	5519	7129	7145
T.DB	003162	2021#	2779*	2798	7123									
T.DCYL	003160	2020#	3043*	3047	3124*	3128	3205*	3209	5296*	5319	5493*	5516	7129	7145
T.DS	003152	2017#												
T.ECPS	003172	2025#												
T.ECPT	003174	2026#	4897*	4898	5010*	5012	5124*	5126	5232*	5233	5425*	5426	5637*	5639
		5762*	5764	5796*	5798	5855*	5856	7140	7148					
T.ER	003154	2018#	2750*	2771	2785*	2795	2894*	3285*	3295	3367*	3377	3456*	3463	3482*
		3568*	3575	3649*	3659	3746*	3753	3849*	3856	3953*	3960	4057*	4064	4163*
		4170	4189*	4289*	4296	4315*	4408*	4415	4434*	4509*	4519	4525*	4599*	4609
		4615*	4689*	4699	4705*	4779*	4789	4795*	5292*	5310	5489*	5507	7116	7120
		7125	7131	7135	7142									
T.MR1	003164	2022#	2886*	2888	2968*	2970	3466*	3467	3756*	3757	3859*	3860	3963*	3964
		4067*	4068	4173*	4174	4299*	4300	4418*	4419	6247*	6248	6261*	6262	6288*
		6289	6301*	6302	7110	7127	7138							
T.MR2	003166	2023#	2245*	2256	2291*	2302	2338*	2349	2384*	2395	2430*	2441	2477*	2488
		2533*	2541	2590*	2598	2644*	2652	2699*	2707	7113				
T.MR3	003170	2024#	2246*	2259	2292*	2305	2339*	2352	2385*	2398	2431*	2444	2478*	2491
		2534*	2544	2591*	2601	2645*	2655	2700*	2710	7113				
T.SPAR	003176	2027#												
T.WC	003142	2013#	2748*	2768	5299*	5316	5496*	5513	7116	7145				
UFE =	000400	1064#												
UNLOAD=	000007	1028#												
UNS =	040000	1089#												
UPE =	020000	1069#												
VRCHDR	053564	3517*	3518	3520*	3523*	3544	7134	8540#						
VV =	000100	1100#												
WCE =	040000	1070#												
WLE =	004000	1086#												
WRDATA=	000023	1034#	2283	2293	2422	2432	2740	2751	2833	2842	2924	3003	3084	3165
		3244	3286	3326	3368	3407	3416	3504	3528	3608	3650	3698	3707	3801
		3810	3905	3914	4009	4018	4114	4123	4240	4249	4359	4368	4463	4510
		4553	4600	4824	4936	5050	5157	5293	5350	5490	5557	5682		
WRDCNT	003264	2062#	2776*	2801*	7123									
WRHEAD=	000027	1036#												
WRL =	004000	1103#												
WRTBIT	034674	4871	4877	4895	4983	4989	5008	5097	5103	5122	5204	5210	5229	5261
		5280	5397	5403	5422	5458	5477	5609	5615	5635	5734	5740	5760	5794
		5830	5849	6228#										
WRTCHK=	000031	1037#	2330	2340	2469	2479	4733	4780						
WRTGAT=	040000	1122#	2887	3761	3864	4870	4982	5096	5203	5396	5608	5733	6228	
SAPTHD	001000	1169	1175#											
SASTAT=	***** U	6374	6389											
SATYC	035556	6345	6347#											
SATY1	035532	6343#												
SATY3	035540	6344#	6480											
SATY4	035550	6346#	6431											
SAUTOB	001134	1206#	2152*	6773	6930									
SBASE	001270	1278#	2163	2171*	2210	2229	2275	2322	2368	2414	2461	2511	2568	2622





\$SVLAD	033506	5955	5994#													
\$SVPC =	000220	1153#	1158													
\$SWR =	167400	867#	878	882	883	884	885	886	887	888	889	1227	1228	1229		
		2109	2110	2112	2113	2228	2274	2321	2367	2413	2460	2510	2567	2621		
		2676	2734	2827	2915	2988	3069	3150	3236	3318	3396	3502	3600	3690		
		3793	3897	4001	4103	4229	4348	4455	4545	4635	4725	4818	4930	5044		
		5150	5343	5542	5670	5866	5872	5899	5905	5907	5936	5937	5938	5939		
		5940	5946	5958	5960	5961	5974	5975	5976	5983	5984	5985	5997	6000		
		6003	6403	6404	6405	6406	6407	6416	6423	6435	6439	6451				
\$SWREG	001236	1250#	2133													
\$SWRMK =	000000	889	890	5940	5941	5964										
\$SWOBT	033560	5970	6004#													
\$TESTN	001220	1241#	5995*	6065*	6066*	7109	7110	7113	7116	7120	7123	7125	7127	7129		
		7131	7135	7138	7140	7142	7148									
\$TIMES	001200	1227#	2109*	2228*	2274*	2321*	2367*	2413*	2460*	2510*	2567*	2621*	2676*	2734*		
		2827*	2915*	2988*	3069*	3150*	3236*	3318*	3396*	3502*	3600*	3690*	3793*	3897*		
		4001*	4103*	4229*	4348*	4455*	4545*	4635*	4725*	4818*	4930*	5044*	5150*	5343*		
		5542*	5670*	5872*	5983*	5990	5993*	6003								
\$TKB	001146	1212#	6678	6699	6710	6735	6763	6790								
\$TKCNT	037134	6679#	6694*	6724	6741*	6855	6857*									
\$TKINT	037144	2086	6694#	6715	6776											
\$TKQEN =	037143	6683#	6749	6860												
\$TKQIN	037136	6680#	6695*	6696	6747*	6748*	6749	6751*								
\$TKQOU	037140	6681#	6696*	6858	6859*	6860	6862*									
\$TKQSR	037142	6682#	6695	6751	6862											
\$TKS	001144	1211#	6678	6700*	6731*	6733	6739*	6761	6777*	6787	6799*	6819*				
\$TKSRV	037214	6697	6710#													
\$TMP0	001160	1219#	2167*	2169	2171	2178*	2180	2182	2194*	2196	2198	2200*	2201*	2202*		
		2203*	2204*	2206												
\$TMP1	001162	1220#														
\$TMP2	001164	1221#														
\$TMP3	001166	1222#														
\$TMP4	001170	1223#	3507*	3518	3520	3523	3579*									
\$TMP5	001172	1224#														
\$TMP6	001174	1225#														
\$TMP7	001176	1226#														
\$TN =	000050	868#	878	2217	2228#	2254	2260	2263	2274#	2300	2306	2309	2321#	2347		
		2353	2356	2367#	2393	2399	2402	2413#	2439	2445	2448	2460#	2486	2492		
		2497	2510#	2555	2567#	2609	2621#	2663	2676#	2715	2721	2734#	2760	2790		
		2809	2827#	2889	2891	2897	2915#	2971	2974	2988#	3055	3069#	3136	3150#		
		3217	3236#	3296	3299	3318#	3378	3381	3396#	3485	3488	3502#	3580	3583		
		3600#	3660	3663	3690#	3763	3766	3793#	3866	3869	3897#	3970	3973	4001#		
		4074	4077	4103#	4192	4195	4229#	4318	4321	4348#	4437	4441	4455#	4528		
		4531	4545#	4618	4621	4635#	4708	4711	4725#	4798	4804	4818#	4907	4930#		
		5021	5044#	5135	5150#	5323	5328	5343#	5520	5529	5542#	5656	5670#	5965		
		6005														
\$TPB	001152	1214#	6520*	6531												
\$TPFLG	001157	1218#	6469	6531												
\$TPS	001150	1213#	6518	6531												
\$TRAP	040756	2104	7069#													
\$TRAP2	041000	7080#	7091													
\$TRP =	000016	7084#	7093#	7094#	7095#	7096#	7097#	7098	7099#	7100	7101#	7102#	7103#	7104#		
		7105#	7106#	7107#												
\$TRPAD	041012	7074	7091#													
\$TSTM	001004	1178#														
\$TSTNM	001102	1191#	5871*	5935	5967*	5994*	5995	6000	6004	6065	6415	6451				









CZR6DCO RK611 DSKLS PRT4  
CZR6DC.P11 10-JUN-80 15:32

MACY11 30A(1052) 10-JUN-80 15:34 PAGE 180  
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0177

.SSAVE	1#	867#	6984
.SSB2D	1#		
.SSB2O	1#		
.SSCOP	1#	867#	5930
.SSIZE	1#	867#	
.SSUPR	1#		
.STRAP	1#	867#	7061
.STYPB	1#		
.STYPD	1#	867#	6608
.STYPE	1#	867#	6452
.STYPO	1#	867#	6531
.\$40CA	1#		
.1170	1#		

. ABS. 055622 000

ERRORS DETECTED: 0

DSKZ:CZR6DC.BIN,DSKZ:CZR6DC.LST/CRF/SOL/NL:TOC=DSKZ:CZR6DC.SML,CZR6DC.P11  
 RUN-TIME: 67 97 8 SECONDS  
 RUN-TIME RATIO: 322/173=1.8  
 CORE USED: 40K (79 PAGES)