

RK611, RK06, RK07

RK611 DSKLS PRT3  
CZR6CD0

AH-9106D-MC  
COPYRIGHT © 76-80  
FICHE 1 OF 1

NOV 1980  
digital  
MADE IN USA

This image shows a microfiche card with a grid of frames. Each frame contains a small, high-contrast image of a document page, likely a technical drawing or data sheet. The frames are arranged in a regular grid pattern across the card. The text within the frames is too small to be legible, but it appears to be organized into columns and rows, possibly representing a table or a series of related diagrams. The overall appearance is that of a standard microfiche used for storing and retrieving digital information.

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31

.REM % IDENTIFICATION

PRODUCT CODE: AC-9104D-MC  
PRODUCT NAME: CZR6CDO RK611 DSKLS PRT3  
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	

32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56

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

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  
112

## 1.0 ABSTRACT

THE RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3

- A. TESTS THE LOADING OF THE DRIVE BUS MESSAGE SHIFT REGISTER FOR CLASS B COMMANDS.
- B. TESTS INDEX AND SECTOR PULSE DETECTION.
- C. TESTS SILO AND NPR TRANSFERS FROM MEMORY IN 16 AND 18 BIT MODE.
- D. TESTS NON-EXISTANT MEMORY AND UNIBUS PARITY ERROR DETECTION.
- E. TESTS READ AND WRITE MFM LOOPBACK.
- F. TESTS CLASS B INSTRUCTION ERRORS.

NO RK06 DRIVE IS REQUIRED FOR PROGRAM EXECUTION.

## 2.0 REQUIREMENTS

### 2.1 EQUIPMENT

PDP-11 SYSTEM (16X 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)

## 3.0 OPERATING PROCEDURES

### 3.1 LOADING PROCEDURE

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

### 3.2 STARTING PROCEDURE

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

### 3.3 OPTIONAL SWITCH SETTINGS

SW15 - HALT PROGRAM  
SW14 - LOOP ON TEST  
SW13 - INHIBIT ERROR TYPE OUT  
SW12 - ABORT AFTER 20 ERRORS  
SW11 - INHIBIT ITERATION COUNT  
SW10 - BELL ON ERROR  
SW9 - LOOP ON ERROR

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  
168

SW8 - LOOP ON TEST IN SWITCHES 0-7

### 3.5 RUN TIME

FIRST PASS	30 SECONDS
SUBSEQUENT PASSES	8:40 MINUTES

### 4.0 OPERATING PROCEDURES

THE PROGRAM IS EXECUTED BY STARTING AT THE APPROPRIATE ADDRESS.

#### 4.1 'SOFTWARE' SWITCH REGISTER

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

SWR = NNNNNN      NEW ='

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

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

#### 4.2 CONTROL C (^C) OPERATION

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

#### 4.3 CONTROL S (^S) OPERATION

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

#### 4.4 CONTROL Q (^Q) OPERATION

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

5.0 PROGRAM DESCRIPTION

\*\*DRIVE MESSAGES FOR CLASS B INSTRUCTIONS

TEST 1 READ HEADER SEEK MESSAGE

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE RK611 CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0 HEAD 0, DRIVE 0. CLOCK IN SEEK MESSAGE INTO SHIFT REGISTER. VERIFY THAT A SEEK IS LOADED WITH THE PROPER BITS IN MESSAGE SET. REPEAT FOR A READ HEADER WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.

TEST 2 WRITE HEADER SEEK MESSAGE

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK IN SEEK MESSAGE INTO SHIFT REGISTER. VERIFY THAT A SEEK IS LOADED WITH THE RTC BIT SET. REPEAT FOR A WRITE HEADER WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.

TEST 3 READ HEADER DRIVE CLEAR MESSAGE

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK SEEK MESSAGE AND MAKE SURE A DRIVE CLEAR IS GENERATED AND THE PROPER BITS ARE SET.

TEST 4 WRITE HEADER DRIVE CLEAR MESSAGE

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER WITH CDT SET IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7. CLOCK SEEK MESSAGE AND LOAD GENERATED DRIVE CLEAR INTO SHIFT REGISTER. MAKE SURE THE DRIVE CLEAR IS GENERATED AND THE PROPER BITS ARE SET.

\*\*INDEX AND SECTOR PULSE DETECT ON

TEST 5 SECTOR PULSE DETECT IN READ HEADER (PART 1)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR MODE, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK ANY DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE, 255 ZEROES AND A ONE.

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  
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  
280

MAKE SURE READ GATE DOES SET.

TEST 6 SECTOR PULSE DETECT IN READ HEADER (PART 2)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR MODE, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE, 255 ZEROES AND A ONE.

MAKE SURE READ GATE DOES NOT SET.

TEST 7 SECTOR PULSE DETECT IN READ HEADER (PART 3)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CHECK MESSAGES. SIMULATE 255 ZEROES AND A ONE.

MAKE SURE READ GATE DOES NOT SET.

TEST 10 INDEX PULSE DETECTION IN WRITE HEADER

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06, IN 26 SECTOR FORMAT, TO CYLINDER 0, HEAD 0, DRIVE 0, WITH A ONE WORD TRANSFER. CLOCK THROUGH THE SEEK AND THE DRIVE CLEAR MESSAGES. ISSUE 200 CONTROLLER CLOCKS AND MAKE SURE WRITE GATE DOES NOT SET. SIMULATE SECTOR PULSE AND 200 CONTROLLER CLOCKS MAKING SURE WRITE GATE DOES NOT SET. SIMULATE INDEX PULSE AND MAKE SURE WRITE GATE SETS.

\*\*NPR READING OF MEMORY

TEST 11 NPR OUTPUT DATA TRANSFER

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 777, HEAD 7, DRIVE 7. SPECIFY A ONE WORD DATA TRANSFER. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE. CLOCK IN FIRST WORD OF NPR TRANSFER. CHECK INPUT READY, OUTPUT READY, BUS ADDRESS, WORD COUNT, AND CONTENTS OF THE SILO. REPEAT FOR 8 DIFFERENT DATA PATTERNS.

TEST 12 PARTIAL SILO FILLING

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER

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  
336

IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. SPECIFY A ONE WORD DATA TRANSFER. CLOCK IN ALL SPECIFIED WORDS INTO THE SILO. CHECK WORD COUNT, BUS ADDRESS, INPUT READY, AND OUTPUT READY. MAKE SURE NO MORE THAN SPECIFIED DATA LENGTH IS CLOCKED INTO THE SILO. CHECK THE SILO FOR CORRECT DATA. REPEAT FOR WORD COUNTS 2-65.

TEST 13 SILO FILLING WITH NPR TRANSFERS

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. SPECIFY A 66 WORD DATA TRANSFER, CLOCK IN ALL 66 WORDS INTO THE SILO. CHECK INPUT READY, OUTPUT READY, BUS ADDRESS, WORD COUNT, AND CONTENTS OF THE SILO.

TEST 14 SILO CAPACITY WITH NPR TRANSFERS

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. SPECIFY A 68 WORD DATA TRANSFER. CLOCK IN 66 WORDS INTO THE SILO. MAKE SURE THAT SILO WILL STOP FILLING AT 66 WORDS. TAKE ONE WORD FROM SILO AND CHECK IT. CLOCK IN NEXT WORD. MAKE SURE NO MORE THAN ONE WORD IS CLOCKED IN THE SILO. TAKE ONE WORD FROM SILO AND CHECK IT. CLOCK IN NEXT WORD. CLOCK IN NEXT WORD. MAKE SURE NO MORE THAN ONE WORD IS CLOCKED IN THE SILO. TAKE ONE WORD FROM SILO AND CHECK IT. ATTEMPT TO CLOCK IN NEXT WORD AND

MAKE SURE NO WORDS ARE CLOCK INTO SILO. UNLOAD THE SILO AND MAKE SURE ALL THE WORDS ARE CORRECT.

TEST 15 BUS ADDRESS INHIBIT

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. SPECIFY A 66 WORD DATA TRANSFER WITH BUS ADDRESS INHIBIT INCREMENT. CHECK WORD COUNT, BUS ADDRESS, INPUT READY, OUTPUT READY, AND MAKE SURE ALL THE WORDS IN THE SILO ARE THE CORRECT SAME WORD.

TEST 16 NON-EXISTENT MEMORY

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06



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  
392

IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
SPECIFY A ONE WORD DATA TO A NON-EXISTENT ADDRESS  
(760000) AND MAKE SURE THE NON-EXISTENT MEMORY ERROR  
OCCURS IN THE RK611 CONTROLLER.

TEST 17 BUS ADDRESS BIT 16

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06  
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
SPECIFY A ONE WORD DATA TRANSFER FROM 200000.  
READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.  
REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 177776.  
CHECK BUS ADDRESS AND WORD COUNT.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 32K  
OF MEMORY IS ON THE SYSTEM.

TEST 20 BUS ADDRESS BIT 17

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06  
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
SPECIFY A ONE WORD DATA TRANSFER FROM 400000.  
READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.  
REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 377776.  
CHECK BUS ADDRESS AND WORD COUNT.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 64K  
OF MEMORY IS ON THE SYSTEM.

TEST 21 ADDRESSING GREATER THAN 96K

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06  
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
SPECIFY A ONE WORD DATA TRANSFER FROM 600000.  
READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.  
REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 577776.  
CHECK BUS ADDRESS AND WORD COUNT.

NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 96K  
OF MEMORY IS ON THE SYSTEM.

TEST 22 SILO FILL IN 18 BIT MODE

CLEAR RK611 WITH CONTROLLER CLEAR. PUT CONTROLLER  
IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06  
IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
SPECIFY 66 WORD DATA TRANSFER. CLOCK

ALL 66 WORD INTO THE SILO. CHECK THAT ALL 66  
WORDS ARE CORRECT (16 LEAST SIGNIFICANT BITS).

\*\*MFM READ LOOPBACK TESTS

TEST 23 READ LOOPBACK (PART 1)

CLEAR RK611 WITH A CONTROLLER CLEAR, PUT CONTROLLER  
IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06  
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.  
SIMULATE SECTOR PULSE, 255 ZEROES, A  
ONE, AND A HEADER CONSISTING OF THE THREE  
FOLLOWING WORDS:

177777  
000000  
177777

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD  
IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

TEST 24 READ LOOPBACK (PART 2)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06  
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.  
SIMULATE SECTOR PULSE, 255 ZEROES, A ONE,  
AND A HEADER CONSISTING OF THE THREE  
FOLLOWING WORDS:

000000  
177777  
000000

MAKE SURE THAT READY COMES UP AFTER THIRD WORD  
IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

TEST 25 READ LOOPBACK (PART 3)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
IN DIAGNOSTIC MOVE. ISSUE A READ HEADER TO AN RK06  
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES  
SIMULATE SECTOR PULSE, 255 ZEROES, A  
ONE, AND A HEADER CONSISTING OF THE THREE  
FOLLOWING WORDS:

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  
448

449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504

125252  
052525  
125252

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

TEST 26 READ LOOPBACK (PART 4)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE, 225 ZEROES, A ONE, AND A HEADER CONSISTING OF THE THREE FOLLOWING WORDS:

044444  
022222  
111111

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

TEST 27 READ LOOPBACK (PART 5)

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE, 255 ZEROES, A ONE, AND A HEADER CONSISTING OF THE THREE FOLLOWING WORDS.

052012  
100520  
052012

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

TEST 30 READ HEADER IN 18 BIT MODE

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE, 255 ZEROES, A ONE, AND A HEADER CONSISTING OF THE THREE FOLLOWING WORDS:

177777  
000000

505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560

177777

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

TEST 31 SYNCH DETECT IN READ HEADER

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE AND 350 ZEROES. MAKE SURE READY REMAINS RESET AND THE SILO REMAINS EMPTY.

TEST 32 ZERO SYNCH ON READ

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE SECTOR PULSE, 255 ZEROES SHIFTED BY A HALF BIT TIME, A ONE, AND A HEADER CONSISTING OF THE THREE FOLLOWING WORDS:

177777  
000000  
177777

MAKE SURE THAT READY COMES AFTER THE THIRD WORD IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

MFM WRITE LOOPBACK TESTS

TEST 33 WRITE ZEROS UNTIL SECTOR PULSE WITH WRITE HEADER

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE AND 500 DATA BITS. MAKE SURE THAT ZEROES ARE WRITTEN. SIMULATE SECTOR PULSE AND MAKE SURE WRITE GATE RESETS.

TEST 34 WRITE LOOPBACK (PART 1)

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

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  
616

177777  
000000  
177777

MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PUL  
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 35 WRITE LOOPBACK (PART 2)

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER

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

000000  
177777  
000000

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD  
IS TRANSFERRED. CHECK FOR CORRECT WRITE ENCODED DATA  
AND PRECOMPENSATION.

TEST 36 WRITE LOOPBACK (PART 3)

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

125252  
052525  
125252

MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PUL  
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 37 WRITE LOOPBACK (PART 4)

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

044444

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  
672

022222  
111111

MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX MOD  
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 40 WRITE LOOPBACK (PART 5)

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

052012  
100520  
052012

MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PUL  
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 41 WRITE LOOPBACK (PART 6)

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

155555  
066666  
155555

MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.  
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 42 WRITE LOOPBACK (PART 7)

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

104210  
104210  
104210

MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.

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  
728

CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION

TEST 43 WRITE TWO HEADERS

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

177777  
000000  
177777

FOLLOW THAT BY A SECTOR PULSE AND ONE THREE WORD HEADER CONSISTING OF THE FOLLOWING DATA:

000000  
177777  
000000

SIMULATE AN INDEX PULSE AND MAKE SURE READY COMES UP. CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

TEST 44 DATA FIELD FILLING ON WRITE HEADER

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, AND SPECIFY TWO 3 WORD HEADERS CONSISTING OF THE FOLLOWING DATA:

125252  
052525  
125252  
052525  
125252  
052525

MAKE SURE THE DATA SYNCH ANY OTHER BITS OF DATA AND ECC FIELD ARE WRITTEN CORRECTLY.

TEST 45 WRITE HEADER FOR 26 SECTORS

CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, SPECIF 66 WORDS. MAKE SURE ALL 26 SECTORS ARE WRITTEN CORRECTLY

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  
784

TEST 46 WRITE HEADER IN 24 SECTOR FORMAT

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER OF SIX WORDS IN 24 SECTOR FORMAT TO AN RK06, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK THROUGH THE SEEK AND DRIVE CLEAR MESSAGES, SIMULATE INDEX PULSE, SECTOR PULSE, 3 HEADER WORDS, SYNCH AND DATA, ANOTHER SECTOR PULSE, 3 HEADER WORDS, AND AN INDEX PULSE. CHECK DATA WRITTEN TO MAKE SURE ONLY LOW 16 BITS OF SILO ARE USED.

\*\*TYPE B INSTRUCTION ERRORS

TEST 47 FORMAT ERROR (PART 1)

CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT TH CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO A RK06, IN 26 SECTOR FORMAT, CYLINDER 43, HEAD 0, DRIVE 0. CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN 0 MAINTENANCE MODE. MAKE SURE FORMAT ERROR, DRIVE AVAILABLE AND CONTROLLER ERROR SET.

TEST 50 FORMAT ERROR (PART 2)

CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT TH CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO A RK06, IN 24 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE 0. CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN 0 MAINTENANCE MODE. MAKE SURE FORMAT ERROR, DRIVE AVAILABLE AND CONTROLLER ERROR SET.

TEST 51 FAULT SETTING CONTROLLER ERROR

CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT TH CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06, IN 26 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE 0. CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN OFF MAINTENANCE MODE. MAKE SURE DRIVE AVAILABLE AND CONTROLLER ERROR SET.

6.0 ERROR REPORTING

THE GENERAL FORMAT OF ERROR REPORTS IS:

OPERATION DESCRIPTION AND ERROR DESCRIPTION

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



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

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
POSTAMBLE	20	1
GAP		
(22(10) SECTOR/TRACK)	160	7
(20(10) SECTOR/TRACK)	140	6

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

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

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

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

x

```
830      ; *** REV 003 ***
831      .NLIST  CND,MD,MC
832      .LIST   ME
833      .ENABL  ABS,AMA
834      $SWR=  167400
835      $TN=    1
836      .TITLE  CZR6CDO RK611 DSKLS CTRL PRT3
837      ;*COPYRIGHT (C) 1976,1980
838      ;*DIGITAL EQUIPMENT CORP.
839      ;*MAYNARD, MASS. 01754
840      ;*
841      ;*
842      ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
843      ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
844      ;*
845      .SBTTL  OPERATIONAL SWITCH SETTINGS
846      ;*
847      ;*      SWITCH      USE
848      ;*      -----      -----
849      ;*      15          HALT ON ERROR
850      ;*      14          LOOP ON TEST
851      ;*      13          INHIBIT ERROR TYPEOUTS
852      ;*      12          ABORT PROGRAM AFTER 20 ERRORS
853      ;*      11          INHIBIT ITERATIONS
854      ;*      10          BELL ON ERROR
855      ;*      9           LOOP ON ERROR
856      ;*      8           LOOP ON TEST IN SWR<7:0>
857      .SBTTL  BASIC DEFINITIONS
858      ;*
859      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
860      STACK= 1100
861      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
862      .EQUIV  IOT,SCOPE     ;;BASIC DEFINITION OF SCOPE CALL
863      ;*
864      ;*MISCELLANEOUS DEFINITIONS
865      HT=    11              ;;CODE FOR HORIZONTAL TAB
866      LF=    12              ;;CODE FOR LINE FEED
867      CR=    15              ;;CODE FOR CARRIAGE RETURN
868      CRLF=  200            ;;CODE FOR CARRIAGE RETURN-LINE FEED
869      PS=    177776         ;;PROCESSOR STATUS WORD
870      .EQUIV  PS,PSW
871      STKLMT= 177774        ;;STACK LIMIT REGISTER
872      PIRQ=   177772        ;;PROGRAM INTERRUPT REQUEST REGISTER
873      DSWR=   177570        ;;HARDWARE SWITCH REGISTER
874      DDISP=  177570       ;;HARDWARE DISPLAY REGISTER
875      ;*
876      ;*GENERAL PURPOSE REGISTER DEFINITIONS
877      R0=     %0             ;;GENERAL REGISTER
878      R1=     %1             ;;GENERAL REGISTER
879      R2=     %2             ;;GENERAL REGISTER
880      R3=     %3             ;;GENERAL REGISTER
881      R4=     %4             ;;GENERAL REGISTER
882      R5=     %5             ;;GENERAL REGISTER
883      R6=     %6             ;;GENERAL REGISTER
884      R7=     %7             ;;GENERAL REGISTER
885      SP=     %6             ;;STACK POINTER
```

167400  
000001

001100

000011  
000012  
000015  
000200  
177776

177774  
177772  
177570  
177570

000000  
000001  
000002  
000003  
000004  
000005  
000006  
000007  
000006

```
886      000007      PC=      %7      ;;PROGRAM COUNTER
887
888      ;*PRIORITY LEVEL DEFINITIONS
889      000000      PR0=      0      ;;PRIORITY LEVEL 0
890      000040      PR1=      40     ;;PRIORITY LEVEL 1
891      000100      PR2=      100    ;;PRIORITY LEVEL 2
892      000140      PR3=      140    ;;PRIORITY LEVEL 3
893      000200      PR4=      200    ;;PRIORITY LEVEL 4
894      000240      PR5=      240    ;;PRIORITY LEVEL 5
895      000300      PR6=      300    ;;PRIORITY LEVEL 6
896      000340      PR7=      340    ;;PRIORITY LEVEL 7
897
898      ;*'SWITCH REGISTER' SWITCH DEFINITIONS
899      100000      SW15=     100000
900      040000      SW14=      40000
901      020000      SW13=      20000
902      010000      SW12=      10000
903      004000      SW11=       4000
904      002000      SW10=       2000
905      001000      SW09=       1000
906      000400      SW08=        400
907      000200      SW07=        200
908      000100      SW06=        100
909      000040      SW05=         40
910      000020      SW04=         20
911      000010      SW03=          10
912      000004      SW02=           4
913      000002      SW01=           2
914      000001      SW00=            1
915      .EQUIV      SW09,SW9
916      .EQUIV      SW08,SW8
917      .EQUIV      SW07,SW7
918      .EQUIV      SW06,SW6
919      .EQUIV      SW05,SW5
920      .EQUIV      SW04,SW4
921      .EQUIV      SW03,SW3
922      .EQUIV      SW02,SW2
923      .EQUIV      SW01,SW1
924      .EQUIV      SW00,SW0
925
926      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
927      100000      BIT15=     100000
928      040000      BIT14=      40000
929      020000      BIT13=      20000
930      010000      BIT12=      10000
931      004000      BIT11=       4000
932      002000      BIT10=       2000
933      001000      BIT09=       1000
934      000400      BIT08=        400
935      000200      BIT07=        200
936      000100      BIT06=        100
937      000040      BIT05=         40
938      000020      BIT04=         20
939      000010      BIT03=          10
940      000004      BIT02=           4
941      000002      BIT01=           2
```

```
942          000001          BIT00= 1
943          .EQUIV BIT09,BIT9
944          .EQUIV BIT08,BIT8
945          .EQUIV BIT07,BIT7
946          .EQUIV BIT06,BIT6
947          .EQUIV BIT05,BIT5
948          .EQUIV BIT04,BIT4
949          .EQUIV BIT03,BIT3
950          .EQUIV BIT02,BIT2
951          .EQUIV BIT01,BIT1
952          .EQUIV BIT00,BIT0
953
954          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
955          000004          ERRVEC= 4          ;;TIME OUT AND OTHER ERRORS
956          000010          RESVEC= 10         ;;RESERVED AND ILLEGAL INSTRUCTIONS
957          000014          TBITVEC=14        ;; "T" BIT
958          000014          TRTVEC= 14         ;;TRACE TRAP
959          000014          BPTVEC= 14         ;;BREAKPOINT TRAP (BPT)
960          000020          IOTVEC= 20         ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
961          000024          PWRVEC= 24         ;;POWER FAIL
962          000030          EMTVEC= 30         ;;EMULATOR TRAP (EMT) **ERROR**
963          000034          TRAPVEC=34        ;; "TRAP" TRAP
964          000060          TKVEC= 60          ;;TTY KEYBOARD VECTOR
965          000064          TPVEC= 64          ;;TTY PRINTER VECTOR
966          000240          PIRQVEC=240       ;;PROGRAM INTERRUPT REQUEST VECTOR
967          .SBTTL MEMORY MANAGEMENT DEFINITIONS
968
969          ;*KT11 VECTOR ADDRESS
970
971          000250          MMVEC= 250
972
973          ;*KT11 STATUS REGISTER ADDRESSES
974
975          177572          SR0= 177572
976          177574          SR1= 177574
977          177576          SR2= 177576
978          172516          SR3= 172516
979
980          ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
981
982          172300          KIPDR0= 172300
983          172302          KIPDR1= 172302
984          172304          KIPDR2= 172304
985          172306          KIPDR3= 172306
986          172310          KIPDR4= 172310
987          172312          KIPDR5= 172312
988          172314          KIPDR6= 172314
989          172316          KIPDR7= 172316
990
991          ;*KERNEL "I" PAGE ADDRESS REGISTERS
992
993          172340          KIPAR0= 172340
994          172342          KIPAR1= 172342
995          172344          KIPAR2= 172344
996          172346          KIPAR3= 172346
997          172350          KIPAR4= 172350
```

```

998      172352      KIPAR5= 172352
999      172354      KIPAR6= 172354
1000     172356      KIPAR7= 172356
1001
1002     000114      MEMVEC= 114          ;MEMORY PARITY VECTOR
1003     172100      MEMBAS= 172100      ;MEM PARITY OPTION
1004     000004      WR.PAR= 4           ;WRITE BAD PARITY
1005     000001      PAR.EN= 1           ;ENABLE PARITY ENABLE
1006     120210      AVECT1= 120210     ;DEFINE RK611 VECTOR ADDRESS
1007     000005      APRIOR= 5           ;DEFINE RK611 PRIORITY
1008     177440      ABASE= 177440      ;DEFINE BASE OF RK611 REGISTERS
1009
1010
1011      .SBTTL  RK611 CONTROLLER REGISTER DEFINITION
1012     000000      RKCS1= 0            ;CONTROL AND STATUS REGISTER 1
1013     000002      RKWC= 2             ;WORD COUNT REGISTER
1014     000004      RKBA= 4             ;BUS ADDRESS REGISTER
1015     000006      RKDA= 6             ;DESIRED TRACK SECTOR REGISTER
1016     000010      RKCS2= 10          ;CONTROL AND STATUS REGISTER 2
1017     000012      RKDS= 12            ;DRIVE STATUS REGISTER
1018     000014      RKER= 14            ;ERROR REGISTER
1019     000016      RKASOF= 16          ;ATTENTION SUMMARY AND OFFSET REGISTER
1020     000020      RKDCYL= 20          ;DESIRED CYLINDER REGISTER
1021     000024      RKDB= 24            ;DATA BUFFER
1022     000026      RKMR1= 26           ;MAINTENANCE REGISTER 1
1023     000034      RKMR2= 34           ;MAINTENANCE REGISTER 2
1024     000036      RKMR3= 36           ;MAINTENANCE REGISTER 3
1025     000030      RKECPS= 30          ;ECC POSITION INFORMATION
1026     000032      RKECPT= 32          ;ECC PATTERN INFORMATION
1027     000022      RKSPAR= 22          ;SPARE REGISTER
1028
1029      .SBTTL  DRIVE COMMANDS
1030
1031     000001      SELDRV= 01          ;SELECT DRIVE
1032     000003      PACK= 03            ;PACK ACKNOWLEDGE
1033     000005      CLEAR= 05           ;DRIVE CLEAR
1034     000007      UNLOAD= 07          ;UNLOAD
1035     000011      SRTSPL= 11          ;START SPINDLE
1036     000013      RECAL= 13           ;RECALIBRATE
1037     000015      OFFSET= 15          ;OFFSET
1038     000017      SEEK= 17            ;SEEK
1039     000021      RDDATA= 21          ;READ DATA
1040     000023      WRDATA= 23          ;WRITE DATA
1041     000025      RDHEAD= 25          ;READ HEADER
1042     000027      WRHEAD= 27          ;WRITE HEADER AND DATA
1043     000031      WRTCHK= 31          ;WRITE CHECK
1044     000300      INTR= 300           ;GENERATE INTERRUPT TO CPU
1045
1046      .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
1047
1048     000001      GO= BIT0            ;GO BIT
1049     000100      IE= BIT6            ;INTERRUPT ENABLE
1050     000200      RDY= BIT7           ;CONTROLLER READY
1051     000400      BA16= BIT8          ;BUS ADDRESS BIT 16
1052     001000      BA17= BIT9          ;BUS ADDRESS BIT 17
1053     002000      CDT= BIT10          ;CONTROLLER DRIVE TYPE (0=RK06)

```

1054	004000	CTO=	BIT11	:CONTROLLER TIMED OUT WAITING FOR
1055				: DRIVE RESPONSE
1056	010000	CFMT=	BIT12	:CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
1057	020000	SPAR=	BIT13	:DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1058	040000	DI=	BIT14	:DRIVE INTERRUPT
1059	100000	CERR=	BIT15	:CONTROLLER ERROR
1060	100000	CCLR=	BIT15	:CONTROLLER CLEAR

.SBTTL CONTROL AND STATUS REGISTER 2 BITS

1061				
1062				
1063				
1064	000007	DRVMSK=	7	:MASK FOR DRIVE SELECTION CODE
1065	000010	P'S=	BIT3	:DESELECT OR RELEASE DRIVE IN BITS 0-2
1066	000020	BAI=	BIT4	:BUS ADDRESS INCREMENT INHIBIT
1067	000040	SCLR=	BIT5	:CLEAR CONTROLLER AND ALL DRIVES
1068	000100	IR=	BIT6	:INPUT READY
1069	000200	OR=	BIT7	:OUTPUT READY
1070	000400	UFE=	BIT8	:UNIT FIELD ERROR
1071	001000	MDS=	BIT9	:MULTIPLE DRIVE SELECT
1072	002000	PGE=	BIT10	:PROGRAMMING ERROR
1073	004000	NEM=	BIT11	:NON-EXISTENT MEMORY
1074	010000	NED=	BIT12	:NON-EXISTENT DRIVE
1075	020000	UPE=	BIT13	:UNIBUS PARITY ERROR
1076	040000	WCE=	BIT14	:WRITE CHECK ERROR
1077	100000	DLT=	BIT15	:DATA LATE ERROR

.SBTTL ERROR REGISTER BIT DEFINITION

1078				
1079				
1080				
1081	000001	ILF=	BIT0	:ILLEGAL FUNCTION CODE
1082	000002	SKI=	BIT1	:SEEK INCOMPLETE
1083	000004	NXF=	BIT2	:NON-EXECUTABLE DRIVE FUNCTION
1084	000010	DRPAR=	BIT3	:DRIVE DETECTED DRIVE BUS PARITY ERROR
1085	000020	FMTE=	BIT4	:FORMAT ERROR
1086	000040	DTYPE=	BIT5	:DRIVE TYPE ERROR
1087	000100	ECH=	BIT6	:ECC HARD
1088	000200	BSE=	BIT7	:BAD SECTOR ERROR
1089	000400	HVRC=	BIT8	:HEADER VRC ERROR
1090	001000	COE=	BIT9	:CYLINDER ADDRESS OVERFLOW ERROR
1091	002000	IDAE=	BIT10	:INVALID DISK ADDRESS ERROR
1092	004000	WLE=	BIT11	:WRITE LOCK ERROR
1093	010000	DTE=	BIT12	:DRIVE TIMING ERROR
1094	020000	OPI=	BIT13	:OPERATION (SEARCH) INCOMPLETE
1095	040000	UNS=	BIT14	:DRIVE UNSAFE
1096	100000	DCK=	BIT15	:DATA CHECK

.SBTTL STATUS REGISTER BIT DEFINITION

1097				
1098				
1099				
1100	000001	DRA=	BIT0	:DRIVE AVAILABLE (CONTROLLER IS SET IF
1101				: THIS BIT IS RESET)
1102	000004	OFST=	BIT2	:DRIVE OFFSET
1103	000010	ACLO=	BIT3	:AC LOW
1104	000020	SPDLSS=	BIT4	:SPEED LOSS
1105	000040	DROT=	BIT5	:DRIVE OFF TRACK
1106	000100	VV=	BIT6	:VOLUME VALID
1107	000200	DRDY=	BIT7	:DRIVE READY
1108	000400	DDT=	BIT8	:DRIVE TYPE (0=RK06)
1109	004000	WRL=	BIT11	:WRITE LOCK

```
1110      020000      PIP=   BIT13      ;POSITIONING IN PROGRESS
1111      040000      DSC=   BIT14      ;DRIVE STATUS CHANGE
1112      100000      SVAL=  BIT15      ;STATUS VALID
1113
1114      .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
1115
1116      000017      MESMSK= 17      ;MESSAGE MASK
1117
1118      000020      PAT=   BIT4      ;FORCE EVEN PARITY ON DRIVE MESSAGE LINES
1119      000040      DMD=   BIT5      ;DIAGNOSTIC MODE
1120      000100      MSP=   BIT6      ;MAINTENANCE SECTOR PULSE
1121      000200      P!ND=  BIT7      ;MAINTENANCE INDEX
1122      000400      MCLK=  BIT8      ;MAINTENANCE CLOCK
1123      001000      MERD=  BIT9      ;MAINTENANCE ENCODED READ DATA
1124      002000      MEWD=  BIT10     ;MAINTENACNE ENCODED WRITE DATA
1125      004000      PCA=   BIT11     ;PRECOMPENSATION ADVANCE
1126      010000      PCD=   BIT12     ;PRECOMPENSATION DELAY
1127      020000      ECCW=  BIT13     ;ECC WORD IS BEING READ OR WRITTEN
1128      040000      WRTGAT= BIT14    ;WRITE GATE
1129      100000      RDGATE= BIT15    ;READ GATE
1130
1131      .SBTTL  TRANSMITTED MESSAGE A
1132
1133      000020      S.SEK=  BIT4      ;SEEK COMMAND
1134      000040      S.RECL= BIT5      ;RECALIBRATE COMMAND
1135      000100      S.STSP= BIT6      ;START SPINDLE COMMAND
1136      000200      S.RTC=  BIT7      ;DRIVE RETURN TO CENTERLINE COMMAND
1137      000400      S.CLR=  BIT8      ;CLEAR ERROR AND DSC
1138      001000      S.FMT=  BIT9      ;FORMAT
1139      002000      S.UNLD= BIT10     ;UNLOAD
1140      004000      S.PACK= BIT11     ;SET VOLUME VALID (PACK ACKNOWLEDGE)
1141
1142      .SBTTL  TRAP CATCHER
1143
1144      .=0
1145      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1146      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1147      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1148      000174      .=174
1149      000174      000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
1150      000176      000000      SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
1151      000200      000137      003662      .SBTTL  STARTING ADDRESS(ES)
1152      000204      000137      003652      JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1153      000214      000214      JMP     RESTRT  ;JUMP TO RESTART ROUTINE
1154      000214      000137      003642      .=214
1155      000214      000137      003642      JMP     PARM    ;JUMP TO OPERATOR ASSIGNED PARMETERS
1156
1157      .SBTTL  ACT11 HOOKS
1158
1159      ;*****
1160      ;HOOKS REQUIRED BY ACT11
1161      000046      000220      $$VPC= .      ;SAVE PC
1162      000052      000046      .=46
1163      000052      042550      $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1164      000052      000052      .=52
1165      000114      000000      .WORD 0      ;;2)SET LOC.52 TO ZERO
1166      000114      000220      .=$$VPC      ;; RESTORE PC
1167      000114      000114      .=MEMVEC
```

1166 000114 043650  
1167 000116 000340  
1168 001000  
1169  
1170  
1171  
1172  
1173  
1174 001000  
1175 000024 000024  
1176 000024 000200  
1177 000044 000044  
1178 000044 001000  
1179 001000  
1180  
1181  
1182  
1183  
1184 001000  
1185 001000 000000  
1186 001002 001214  
1187 001004 000000  
1188 001006 000000  
1189 001010 000000  
1190 001012 000032

MEMERR  
PR7  
.=1000  
.SBTTL APT PARAMETER BLOCK  
:\*\*\*\*\*  
:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT  
:\*\*\*\*\*  
.\$X=. ;;SAVE CURRENT LOCATION  
.=24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM  
200 ;;FOR APT START UP  
.=44 ;;POINT TO APT INDIRECT ADDRESS PNTR.  
\$APTHDR ;;POINT TO APT HEADER BLOCK  
.=.\$X ;;RESET LOCATION COUNTER  
:\*\*\*\*\*  
:SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC  
:INTERFACE SPEC.  
\$APTHD:  
\$HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.  
\$MBADR: .WORD \$MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)  
\$STMT: .WORD ;;RUN TIM OF LONGEST TEST  
\$PASTM: .WORD ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)  
\$UNITM: .WORD ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT  
SETEND-\$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)



1191  
1192  
1193  
1194  
1195  
1196  
1197 001100  
1198 001100  
1199 001100 000000  
1200 001102 000  
1201 001103 000  
1202 001104 000000  
1203 001106 000000  
1204 001110 000000  
1205 001112 000000  
1206 001114 000  
1207 001115 001  
1208 001116 000000  
1209 001120 000000  
1210 001122 000000  
1211 001124 000000  
1212 001126 000000  
1213 001130 000000  
1214 001132 000000  
1215 001134 000  
1216 001135 000  
1217 001136 000000  
1218 001140 177570  
1219 001142 177570  
1220 001144 177560  
1221 001146 177562  
1222 001150 177564  
1223 001152 177566  
1224 001154 000  
1225 001155 002  
1226 001156 012  
1227 001157 000  
1228 001160 000000  
1229 001162 000000  
1230 001164 000000  
1231 001166 000000  
1232 001170 000000  
1233 001172 000000  
1234 001174 000000  
1235 001176 000000  
1236 001200 000000  
1237 001202 000000  
1238 001204 177607 000377  
1239 001210 077  
1240 001211 015  
1241 001212 000012  
1242  
1243  
1244  
1245  
1246

```
.SBTTL COMMON TAGS

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

SCMTAG:      .=1100                ;;START OF COMMON TAGS
              .WORD 0
STSTNM:      .BYTE 0                ;;CONTAINS THE TEST NUMBER
SERFLG:      .BYTE 0                ;;CONTAINS ERROR FLAG
S!CNT:       .WORD 0                ;;CONTAINS SUBTEST ITERATION COUNT
SLPADR:      .WORD 0                ;;CONTAINS SCOPE LOOP ADDRESS
SLPERR:      .WORD 0                ;;CONTAINS SCOPE RETURN FOR ERRORS
SERTTL:      .WORD 0                ;;CONTAINS TOTAL ERRORS DETECTED
SITEMB:      .BYTE 0                ;;CONTAINS ITEM CONTROL BYTE
SERMAX:      .BYTE 1                ;;CONTAINS MAX. ERRORS PER TEST
SERRPC:      .WORD 0                ;;CONTAINS PC OF LAST ERROR INSTRUCTION
SGDADR:      .WORD 0                ;;CONTAINS ADDRESS OF 'GOOD' DATA
SBDADR:      .WORD 0                ;;CONTAINS ADDRESS OF 'BAD' DATA
SGDDAT:      .WORD 0                ;;CONTAINS 'GOOD' DATA
SBDDAT:      .WORD 0                ;;CONTAINS 'BAD' DATA
              .WORD 0                ;;RESERVED--NOT TO BE USED
              .WORD 0
SAUTOB:      .BYTE 0                ;;AUTOMATIC MODE INDICATOR
SINTAG:      .BYTE 0                ;;INTERRUPT MODE INDICATOR
              .WORD 0
SWR:         .WORD DSWR              ;;ADDRESS OF SWITCH REGISTER
DISPLAY:     .WORD DDISP             ;;ADDRESS OF DISPLAY REGISTER
$TKS:       177560                    ;;TTY KBD STATUS
$TKB:       177562                    ;;TTY KBD BUFFER
$TPS:       177564                    ;;TTY PRINTER STATUS REG. ADDRESS
$TPB:       177566                    ;;TTY PRINTER BUFFER REG. ADDRESS
$NULL:      .BYTE 0                ;;CONTAINS NULL CHARACTER FOR FILLS
$FILLS:     .BYTE 2                ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
$FILLC:     .BYTE 12               ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
$TPFLG:     .BYTE 0                ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
$TMP0:      .WORD 0                ;;USER DEFINED
$TMP1:      .WORD 0                ;;USER DEFINED
$TMP2:      .WORD 0                ;;USER DEFINED
$TMP3:      .WORD 0                ;;USER DEFINED
$TMP4:      .WORD 0                ;;USER DEFINED
$TMP5:      .WORD 0                ;;USER DEFINED
$TMP6:      .WORD 0                ;;USER DEFINED
$TMP7:      .WORD 0                ;;USER DEFINED
$TIMES:     0                        ;;MAX. NUMBER OF ITERATIONS
$ESCAPE:    0                        ;;ESCAPE ON ERROR ADDRESS
$BELL:      .ASCII <207><377><377> ;;CODE FOR BELL
$QUES:      .ASCII /?/              ;;QUESTION MARK
$CRLF:      .ASCII <15>             ;;CARRIAGE RETURN
$LF:        .ASCII <12>            ;;LINE FEED

*****
.SBTTL APT MAILBOX-ETABLE

*****
.EVEN
```

1247	001214		SMAIL:		::APT MAILBOX	
1248	001214	000000	SMSGTY:	.WORD	AMSGTY	::MESSAGE TYPE CODE
1249	001216	000000	SFATAL:	.WORD	AFATAL	::FATAL ERROR NUMBER
1250	001220	000000	\$TESTN:	.WORD	ATESTN	::TEST NUMBER
1251	001222	000000	\$PASS:	.WORD	APASS	::PASS COUNT
1252	001224	000000	\$DEVCT:	.WORD	ADEVCT	::DEVICE COUNT
1253	001226	000000	\$UNIT:	.WORD	AUNIT	::I/O UNIT NUMBER
1254	001230	000000	\$MSGAD:	.WORD	AMSGAD	::MESSAGE ADDRESS
1255	001232	000000	\$MSGLG:	.WORD	AMSGLG	::MESSAGE LENGTH
1256	001234		SETABLE:			::APT ENVIRONMENT TABLE
1257	001234	000	SENV:	.BYTE	AENV	::ENVIRONMENT BYTE
1258	001235	000	SENVN:	.BYTE	AENVN	::ENVIRONMENT MODE BITS
1259	001236	000000	\$SWREG:	.WORD	ASWREG	::APT SWITCH REGISTER
1260	001240	000000	\$USWR:	.WORD	AUSWR	::USER SWITCHES
1261	001242	000000	\$CPUOP:	.WORD	ACPUOP	::CPU TYPE,OPTIONS
1262			::*			BITS 15-11=CPU TYPE
1263			::*			11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1264			::*			11/70=06,PDQ=07,Q=10
1265			::*			BIT 10=REAL TIME CLOCK
1266			::*			BIT 9=FLOATING POINT PROCESSOR
1267			::*			BIT 8=MEMORY MANAGEMENT
1268	001244	000	\$MAMS1:	.BYTE	AMAMS1	::HIGH ADDRESS,M.S. BYTE
1269	001245	000	\$MTYP1:	.BYTE	AMTYP1	::MEM. TYPE,BLK#1
1270			::*			MEM.TYPE BYTE -- (HIGH BYTE)
1271			::*			900 NSEC CORE=001
1272			::*			300 NSEC BIPOLAR=002
1273			::*			500 NSEC MOS=003
1274	001246	000000	\$MADR1:	.WORD	AMADR1	::HIGH ADDRESS,BLK#1
1275			::*			MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1276	001250	000	\$MAMS2:	.BYTE	AMAMS2	::HIGH ADDRESS,M.S. BYTE
1277	001251	000	\$MTYP2:	.BYTE	AMTYP2	::MEM. TYPE,BLK#2
1278	001252	000000	\$MADR2:	.WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
1279	001254	000	\$MAMS3:	.BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
1280	001255	000	\$MTYP3:	.BYTE	AMTYP3	::MEM. TYPE,BLK#3
1281	001256	000000	\$MADR3:	.WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
1282	001260	000	\$MAMS4:	.BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
1283	001261	000	\$MTYP4:	.BYTE	AMTYP4	::MEM. TYPE,BLK#4
1284	001262	000000	\$MADR4:	.WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
1285	001264	120210	\$VECT1:	.WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
1286	001266	000000	\$VECT2:	.WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
1287	001270	177440	\$BASE:	.WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
1288	001272	000000	\$DEVN:	.WORD	ADEVN	::DEVICE MAP
1289	001274	000000	\$CDW1:	.WORD	ACDW1	::CONTROLLER DESCRIPTION WORD#1
1290	001276	000000	\$CDW2:	.WORD	ACDW2	::CONTROLLER DESCRIPTION WORD#2
1291	001300		SETEND:			
1292			.MEXIT			

1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307 001300  
1308  
1309  
1310 001300 055321  
1311 001302 061500  
1312 001304 050560  
1313 001306 051222  
1314  
1315  
1316 001310 055321  
1317 001312 061543  
1318 001314 050560  
1319 001316 051222  
1320  
1321  
1322 001320 055321  
1323 001322 061567  
1324 001324 050560  
1325 001326 051222  
1326  
1327  
1328 001330 055403  
1329 001332 061500  
1330 001334 050560  
1331 001336 051222  
1332  
1333  
1334 001340 055403  
1335 001342 061543  
1336 001344 050560  
1337 001346 051222  
1338  
1339  
1340 001350 055403  
1341 001352 061567  
1342 001354 050560  
1343 001356 051222  
1344  
1345  
1346 001360 055466  
1347 001362 061500  
1348 001364 050560

.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: ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER  
CS1 INCORRECT.  
EM200  
EM3000  
DT001  
DF001  
ERROR 2: ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER  
MESSAGE A INCORRECT.  
EM200  
EM3001  
DT001  
DF001  
ERROR 3: ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER  
MESSAGE B INCORRECT.  
EM200  
EM3002  
DT001  
DF001  
ERROR 4: ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER  
CS1 INCORRECT.  
EM201  
EM3000  
DT001  
DF001  
ERROR 5: ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER  
MESSAGE A INCORRECT.  
EM201  
EM3001  
DT001  
DF001  
ERROR 6: ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER  
MESSAGE IS INCORRECT.  
EM201  
EM3002  
DT001  
DF001  
ERROR 7: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER  
CS1 INCORRECT  
EM202  
EM3000  
DT001

1349	001366	051222	DF001
1350	:	:	ERROR 10: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
1351	:	:	MESSAGE A INCORRECT.
1352	001370	055466	EM202
1353	001372	061543	EM3001
1354	001374	050560	DT001
1355	001376	051222	DF001
1356	:	:	ERROR 11: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
1357	:	:	MESSAGE B INCORRECT.
1358	001400	055466	EM202
1359	001402	061567	EM3002
1360	001404	050560	DT001
1361	001406	051222	DF001
1362	:	:	ERROR 12: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1363	:	:	CS1 INCORRECT.
1364	001410	055557	EM203
1365	001412	061500	EM3000
1366	001414	050560	DT001
1367	001416	051222	DF001
1368	:	:	ERROR 13: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1369	:	:	MESSAGE A INCORRECT.
1370	001420	055557	EM203
1371	001422	061543	EM3001
1372	001424	050560	DT001
1373	001426	051222	DF001
1374	:	:	ERROR 14: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1375	:	:	MESSAGE B INCORRECT.
1376	001430	055557	EM203
1377	001432	061567	EM3002
1378	001434	050560	DT001
1379	001436	051222	DF001
1380	:	:	ERROR 15: ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1381	:	:	CS1 INCORRECT AFTER SENDING DRIVE CLEAR.
1382	001440	055651	EM204
1383	001442	061613	EM3003
1384	001444	050600	DT015
1385	001446	051246	DF015
1386	:	:	ERROR 16: ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1387	:	:	CS1 INCORRECT AFTER DATE SIMULATION.
1388	001450	055651	EM204
1389	001452	061663	EM3004
1390	001454	050600	DT015
1391	001456	051246	DF015
1392	:	:	ERROR 17: ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1393	:	:	MAINT REG. 1 INCORRECT DURING DATA SIMULATION (SECTOR PULSE)
1394	001460	055651	EM204
1395	001462	061735	EM3005
1396	001464	050610	DT017
1397	001466	051272	DF017
1398	:	:	ERROR 20: ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1399	:	:	MAINT REG. 1 INCORRECT DURING DATA SIMULATION (INDEX PULSE)
1400	001470	055651	EM204
1401	001472	062037	EM3006
1402	001474	050610	DT017
1403	001476	051272	DF017
1404	:	:	ERROR 21: ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT

1405	:		MAINT REG. 1 INCORRECT DURING DATA SIMULATION (NO INDEX OR SECTOR)
1406	001500	055651	EM204
1407	001502	062140	EM3007
1408	001504	050610	DT017
1409	001506	051272	DF017
1410	:	:	ERROR 22: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1411	:	:	CS1 INCORRECT AFTER SENDING DRIVE CLEAR
1412	001510	055737	EM205
1413	001512	061613	EM3003
1414	001514	050626	DT022
1415	001516	051316	DF022
1416	:	:	ERROR 23: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1417	:	:	BUS ADD INCORRECT AFTER SENDING DRIVE CLEAR.
1418	001520	055737	EM205
1419	001522	062261	EM3008
1420	001524	050626	DT022
1421	001526	051316	DF022
1422	:	:	ERROR 24: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1423	:	:	WORD COUNT INCORRECT AFTER SENDING DRIVE CLEAR
1424	001530	055737	EM205
1425	001532	062341	EM3009
1426	001534	050626	DT022
1427	001536	051316	DF022
1428	:	:	ERROR 25: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1429	:	:	MAINT REG 1 INCORRECT DURING DATA SIMULATION (NO INDEX OR SECTOR)
1430	001540	055737	EM205
1431	001542	062637	EM3014
1432	001544	050646	DT025
1433	001546	051342	DF025
1434	:	:	ERROR 26: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1435	:	:	CS1 CHANGED DURING COMMAND EXECUTION
1436	001550	055737	EM205
1437	001552	062420	EM3010
1438	001554	050626	DT022
1439	001556	051316	DF022
1440	:	:	ERROR 27: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1441	:	:	BUS ADDRESS CHANGED BEFORE INDEX PULSE
1442	001560	055737	EM205
1443	001562	062465	EM3011
1444	001564	050626	DT022
1445	001566	051316	DF022
1446	:	:	ERROR 30: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1447	:	:	WORD COUNT CHANGED BEFORE INDEX PULSE
1448	001570	055737	EM205
1449	001572	062534	EM3012
1450	001574	050626	DT022
1451	001576	051316	DF022
1452	:	:	ERROR 31: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1453	:	:	CS1 CHANGED AFTER INDEX PULSE
1454	001600	055737	EM205
1455	001602	062602	EM3013
1456	001604	050660	DT031
1457	001606	051366	DF031
1458	:	:	ERROR 32: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1459	:	:	BUS ADDRESS CHANGED AFTER INDEX PULSE.
1460	001610	055737	EM205

1461	001612	062710	EM3015
1462	001614	050660	DT031
1463	001616	051366	DF031
1464	:	:	ERROR 33: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1465	:	:	WORD COUNT CHANGED AFTER INDEX PULSE
1466	001620	055737	EM205
1467	001622	062756	EM3016
1468	001624	050660	DT031
1469	001626	051366	DF031
1470	:	:	ERROR 34: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1471	:	:	MAINT REG 1 INCORRECT AFTER SECTOR PULSE
1472	001630	055737	EM205
1473	001632	063336	EM3025
1474	001634	050660	DT031
1475	001636	051366	DF031
1476	:	:	ERROR 35: ATTEMPTING AN NPR READ OF ONE WORD
1477	:	:	CS1 INCORRECT
1478	001640	056025	EM206
1479	001642	061500	EM3000
1480	001644	050704	DT035
1481	001646	051412	DF035
1482	:	:	ERROR 36: ATTEMPTING AN NPR READ OF ONE WORD
1483	:	:	CS2 INCORRECT
1484	001650	056025	EM206
1485	001652	063023	EM3018
1486	001654	050704	DT035
1487	001656	051412	DF035
1488	:	:	ERROR 37: ATTEMPTING AN NPR READ OF ONE WORD
1489	:	:	BIJS ADDRESS INCORRECT
1490	001660	056025	EM206
1491	001662	063067	EM3019
1492	001664	050704	DT035
1493	001666	051412	DF035
1494	:	:	ERROR 40: ATTEMPTING AN NPR READ OF ONE WORD
1495	:	:	WORD COUNT REG INCORRECT
1496	001670	056025	EM206
1497	001672	063115	EM3020
1498	001674	050704	DT035
1499	001676	051412	DF035
1500	:	:	ERROR 41: ATTEMPTING AN NPR READ OF ONE WORD
1501	:	:	WORD READ INCORRECT
1502	001700	056025	EM206
1503	001702	063146	EM3021
1504	001704	050730	DT041
1505	001706	051436	DF041
1506	:	:	ERROR 42: ATTEMPTING AN NPR READ OF ONE WORD
1507	:	:	CS1 INCORRECT AFTER READING DATA BUFFER
1508	001710	056025	EM206
1509	001712	063172	EM3022
1510	001714	050740	DT042
1511	001716	051462	DF042
1512	:	:	ERROR 43: ATTEMPTING AN NPR READ OF ONE WORD
1513	:	:	CS2 INCORRECT AFTER READING DATA BUFFER
1514	001720	056025	EM206
1515	001722	063242	EM3023
1516	001724	050740	DT042

1517	001726	051462	DF042	
1518			ERROR 44:	ATTEMPTING AN NPR READ OF ONE WORD
1519				CS1 INCORRECT
1520	001730	056070	EM207	
1521	001732	061500	EM3000	
1522	001734	050704	DT035	
1523	001736	051412	DF035	
1524			ERROR 45:	ATTEMPTING AN NPR READ
1525				CS2 INCORRECT
1526	001740	056070	EM207	
1527	001742	063023	EM3018	
1528	001744	050704	DT035	
1529	001746	051412	DF035	
1530			ERROR 46:	ATTEMPTING AN NPR READ
1531				BUS ADDRESS INCORRECT
1532	001750	056070	EM207	
1533	001752	063067	EM3019	
1534	001754	050704	DT035	
1535	001756	051412	DF035	
1536			ERROR 47:	ATTEMPTING AN NPR READ
1537				WORD COUNT INCORRECT
1538	001760	056070	EM207	
1539	001762	063115	EM3020	
1540	001764	050704	DT035	
1541	001766	051412	DF035	
1542			ERROR 50:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1543				CS1 INCORRECT
1544	001770	056117	EM208	
1545	001772	061500	EM3000	
1546	001774	050704	DT035	
1547	001776	051412	DF035	
1548			ERROR 51:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1549				CS2 INCORRECT
1550	002000	056117	EM208	
1551	002002	063023	EM3018	
1552	002004	050704	DT035	
1553	002006	051412	DF035	
1554			ERROR 52:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1555				BUS ADDRESS INCORRECT
1556	002010	056117	EM208	
1557	002012	063067	EM3019	
1558	002014	050704	DT035	
1559	002016	051412	DF035	
1560			ERROR 53:	ATTEMPTING NPR READ CHECKING ZERO DETECT
1561				WORD COUNT INCORRECT
1562	002020	056117	EM208	
1563	002022	063115	EM3020	
1564	002024	050704	DT035	
1565	002026	051412	DF035	
1566			ERROR 54:	ATTEMPTING NPR READ
1567				DATA BUFFER INCORRECT
1568	002030	056070	EM207	
1569	002032	063146	EM3021	
1570	002034	050754	DT054	
1571	002036	051506	DF054	
1572			ERROR 55:	ATTEMPTING NPR READ

1573			:		CS1 INCORRECT AFTER READING DATA BUFFER
1574	002040	056070	:	EM207	
1575	002042	063172	:	EM3022	
1576	002044	050766	:	DT055	
1577	002046	051532	:	DF055	
1578			:	ERROR 56:	ATTEMPTING NPR READ
1579			:		CS2 INCORRECT AFTER READING DATA BUFFER
1580	002050	056070	:	EM207	
1581	002052	063242	:	EM3023	
1582	002054	050766	:	DT055	
1583	002056	051532	:	DF055	
1584			:	ERROR 57:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1585			:		INHIBIT CS1 INCORRECT
1586	002060	056170	:	EM209	
1587	002062	061500	:	EM3000	
1588	002064	050704	:	DT035	
1589	002066	051412	:	DF035	
1590			:	ERROR 60:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1591			:		INHIBIT CS2 INCORRECT
1592	002070	056170	:	EM209	
1593	002072	063023	:	EM3018	
1594	002074	050704	:	DT035	
1595	002076	051412	:	DF035	
1596			:	ERROR 61:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1597			:		INHIBIT BUS ADDRESS INCORRECT
1598	002100	056170	:	EM209	
1599	002102	063067	:	EM3019	
1600	002104	050704	:	DT035	
1601	002106	051412	:	DF035	
1602			:	ERROR 62:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1603			:		INHIBIT WORD COUNT INCORRECT
1604	002110	056170	:	EM209	
1605	002112	063115	:	EM3020	
1606	002114	050704	:	DT035	
1607	002116	051412	:	DF035	
1608			:	ERROR 63:	ATTEMPTING NPR READ WITH IBA TO CHECK ZERO
1609			:		DETECT-CS1 INCORRECT
1610	002120	056257	:	EM210	
1611	002122	061500	:	EM3000	
1612	002124	050704	:	DT035	
1613	002126	051412	:	DF035	
1614			:	ERROR 64:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1615			:		DETECT-CS2 INCORRECT
1616	002130	056257	:	EM210	
1617	002132	063023	:	EM3018	
1618	002134	050704	:	DT035	
1619	002136	051412	:	DF035	
1620			:	ERROR 65:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1621			:		DETECT-BUS ADDRESS INCORRECT
1622	002140	056257	:	EM210	
1623	002142	063067	:	EM3019	
1624	002144	050704	:	DT035	
1625	002146	051412	:	DF035	
1626			:	ERROR 66:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1627			:		DETECT-WORD COUNT INCORRECT
1628	002150	056257	:	EM210	



1629	002152	063115	EM3020
1630	002154	050704	DT035
1631	002156	051412	DF035
1632	:	:	ERROR 67: ATTEMPTING NPR READ WITH BUS ADDRESS INHIBIT
1633	:	:	INCREMENT-DATA BUFFER INCORRECT
1634	002160	056170	EM209
1635	002162	063146	EM3021
1636	002164	050754	DT054
1637	002166	051506	DF054
1638	:	:	ERROR 70: ATTEMPTING NPR READ WITH BUS ADDRESS INHIBIT
1639	:	:	INCREMENT-CS1 INCORRECT AFTER READING DATA BUFFER
1640	002170	056170	EM209
1641	002172	063172	EM3022
1642	002174	050766	DT055
1643	002176	051532	DF055
1644	:	:	ERROR 71: ATTEMPTING NPR READ WITH ADDRESS BUFFER INHIBIT
1645	:	:	INCREMENT-CS2 INCORRECT AFTER READING DATA BUFFER
1646	002200	056170	EM209
1647	002202	063242	EM3023
1648	002204	050766	DT055
1649	002206	051532	DF055
1650	:	:	ERROR 72: ATTEMPTING TO FORCE NON-EXISTANT MEMORY
1651	:	:	CS1 INCORRECT
1652	002210	056373	EM211
1653	002212	061500	EM3000
1654	002214	051004	DT072
1655	002216	051556	DF072
1656	:	:	ERROR 73: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1657	:	:	CS2 INCORRECT
1658	002220	056373	EM211
1659	002222	063023	EM3018
1660	002224	051004	DT072
1661	002226	051556	DF072
1662	:	:	ERROR 74: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1663	:	:	ERROR REG INCORRECT
1664	002230	056373	EM211
1665	002232	063312	EM3024
1666	002234	051004	DT072
1667	002236	051556	DF072
1668	:	:	ERROR 75: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1669	:	:	BUS ADDRESS INCORRECT
1670	002240	056373	EM211
1671	002242	063067	EM3019
1672	002244	051004	DT072
1673	002246	051556	DF072
1674	:	:	ERROR 76: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1675	:	:	WORD COUNT INCORRECT
1676	002250	056373	EM211
1677	002252	063115	EM3020
1678	002254	051004	DT072
1679	002256	051556	DF072
1680	:	:	ERROR 77: ATTEMPTING TO CLEAR NON-EXISTENT MEMORY
1681	:	:	CS1 INCORRECT
1682	002260	056443	EM212
1683	002262	061500	EM3000
1684	002264	051034	DT077

1685	002266	051612	DF077
1686			ERROR 100: ATTEMPTING TO CLEAR NON-EXISTENT MEMORY
1687			CS2 INCORRECT
1688	002270	056443	EM212
1689	002272	063023	EM3018
1690	002274	051034	DT077
1691	002276	051612	DF077
1692			ERROR 101: TESTING EXTENDED MEMORY ADDRESSING BITS
1693			CS1 INCORRECT
1694	002300	056513	EM213
1695	002302	061500	EM3000
1696	002304	050704	DT035
1697	002306	051412	DF035
1698			ERROR 102: TESTING EXTENDED MEMORY ADDRESSING BITS
1699			CS2 INCORRECT
1700	002310	056513	EM213
1701	002312	063023	EM3018
1702	002314	050704	DT035
1703	002316	051412	DF035
1704			ERROR 103: TESTING EXTENDED MEMORY ADDRESSING BITS
1705			BUS ADDRESS INCORRECT
1706	002320	056513	EM213
1707	002322	063067	EM3019
1708	002324	050704	DT035
1709	002326	051412	DF035
1710			ERROR 104: TESTING EXTEND MEMORY ADDRESSING BITS
1711			WORD COUNT INCORRECT
1712	002330	056513	EM213
1713	002332	063115	EM3020
1714	002334	050704	DT035
1715	002336	051412	DF035
1716			ERROR 105: TESTING EXTENDED MEMORY ADDRESSING BITS
1717			DATA BUFFER INCORRECT
1718	002340	056513	EM213
1719	002342	063146	EM3021
1720	002344	050730	DT041
1721	002346	051436	DF041
1722			ERROR 106: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1723			CS1 INCORRECT
1724	002350	056563	EM214
1725	002352	061500	EM3000
1726	002354	051004	DT072
1727	002356	051556	DF072
1728			ERROR 107: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1729			CS2 INCORRECT
1730	002360	056563	EM214
1731	002362	063023	EM3018
1732	002364	051004	DT072
1733	002366	051556	DF072
1734			ERROR 110: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1735			BUS ADDRESS INCORRECT
1736	002370	056563	EM214
1737	002372	063067	EM3019
1738	002374	051004	DT072
1739	002376	051556	DF072
1740			ERROR 111: ATEMPTING TO FORCE UNIBUS PARITY ERROR

1741			:	WORD COUNT INCORRECT
1742	002400	056563	:	EM214
1743	002402	063115	:	EM3020
1744	002404	051004	:	DT072
1745	002406	051556	:	DF072
1746			:	ERROR 112: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1747			:	CS1 INCORRECT
1748	002410	056633	:	EM215
1749	002412	061500	:	EM3000
1750	002414	050704	:	DT035
1751	002416	051412	:	DF035
1752			:	ERROR 113: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1753			:	CS2 INCORRECT
1754	002420	056633	:	EM215
1755	002422	063023	:	EM3018
1756	002424	050704	:	DT035
1757	002426	051412	:	DF035
1758			:	ERROR 114: ATTEMPTING NPR READ OR LOCATION PRIOR TO BAD PARITY
1759			:	BUS ADDRESS INCORRECT
1760	002430	056633	:	EM215
1761	002432	063067	:	EM3019
1762	002434	050704	:	DT035
1763	002436	051412	:	DF035
1764			:	ERROR 115: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1765			:	WORD COUNT INCORRECT
1766	002440	056633	:	EM215
1767	002442	063023	:	EM3018
1768	002444	050704	:	DT035
1769	002446	051412	:	DF035
1770			:	ERROR 116: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1771			:	ERROR REG INCORRECT
1772	002450	056563	:	EM214
1773	002452	063312	:	EM3024
1774	002454	051004	:	DT072
1775	002456	051556	:	DF072
1776			:	ERROR 117: ATTEMPTING TO CLEAR UNIBUS PARITY ERROR
1777			:	CS1 INCORRECT
1778	002460	056717	:	EM216
1779	002462	061500	:	EM3000
1780	002464	051034	:	DT077
1781	002466	051612	:	DF077
1782			:	ERROR 120: ATTEMPTING TO CLEAR UNIBUS PARITY ERROR
1783			:	CS2 INCORRECT
1784	002470	056717	:	EM216
1785	002472	063023	:	EM3018
1786	002474	051034	:	DT077
1787	002476	051612	:	DF077
1788			:	ERROR 121: ATTEMPTING 18 BIT NPR READ
1789			:	CS1 INCORRECT
1790	002500	056767	:	EM217
1791	002502	061500	:	EM3000
1792	002504	050704	:	DT035
1793	002506	051412	:	DF035
1794			:	ERROR 122: ATTEMPTING 18 BIT NPR READ
1795			:	CS2 INCORRECT
1796	002510	056767	:	EM217

1797	002512	063023	EM3018
1798	002514	050704	DT035
1799	002516	051412	DF035
1800	:	:	ERROR 123: ATTEMPTING 18 BIT NPR READ
1801	:	:	BUS ADDRESS INCORRECT
1802	002520	056767	EM217
1803	002522	063067	EM3019
1804	002524	050704	DT035
1805	002526	051412	DF035
1806	:	:	ERROR 124: ATTEMPTING 18 BIT NPR READ
1807	:	:	WORD COUNT INCORRECT
1808	002530	056767	EM217
1809	002532	063115	EM3020
1810	002534	050704	DT035
1811	002536	051412	DF035
1812	:	:	ERROR 125: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1813	:	:	CS1 INCORRECT
1814	002540	057022	EM218
1815	002542	061500	EM3000
1816	002544	050704	DT035
1817	002546	051412	DF035
1818	:	:	ERROR 126: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1819	:	:	CS2 INCORRECT
1820	002550	057022	EM218
1821	002552	063023	EM3018
1822	002554	050704	DT035
1823	002556	051412	DF035
1824	:	:	ERROR 127: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1825	:	:	BUS ADDRESS INCORRECT
1826	002560	057022	EM218
1827	002562	063067	EM3019
1828	002564	050704	DT035
1829	002566	051412	DF035
1830	:	:	ERROR 130: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1831	:	:	WORD COUNT INCORRECT
1832	002570	057022	EM218
1833	002572	063115	EM3020
1834	002574	050704	DT035
1835	002576	051412	DF035
1836	:	:	ERROR 131: ATTEMPTING 18 BIT NPR READ
1837	:	:	DATA BUFFER INCORRECT
1838	002600	056767	EM217
1839	002602	063146	EM3021
1840	002604	050740	DT042
1841	002606	051462	DF042
1842	:	:	ERROR 132: ATTEMPTING 18 BIT NPR READ
1843	:	:	CS1 INCORRECT AFTER READING DATA BUFFER
1844	002610	056767	EM217
1845	002612	063172	EM3022
1846	002614	050766	DT055
1847	002616	051532	DF055
1848	:	:	ERROR 133: ATTEMPTING 18 BIT NPR READ
1849	:	:	CS2 INCORRECT AFTER READING DATA BUFFER
1850	002620	056767	EM217
1851	002622	063242	EM3023
1852	002624	050766	DT055

1853	002626	051532	DF055
1854			ERROR 134: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1855			CS1 INCORRECT
1856	002630	057102	EM219
1857	002632	061500	EM3000
1858	002634	050704	DT035
1859	002636	051412	DF035
1860			ERROR 135: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1861			CS2 INCORRECT
1862	002640	057102	EM219
1863	002642	063023	EM3018
1864	002644	050704	DT035
1865	002646	051412	DF035
1866			ERROR 136: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1867			BUS ADDRESS INCORRECT
1868	002650	057102	EM219
1869	002652	063067	EM3019
1870	002654	050704	DT035
1871	002656	051412	DF035
1872			ERROR 137: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1873			WORD COUNT INCORRECT
1874	002660	057102	EM219
1875	002662	063115	EM3020
1876	002664	050704	DT035
1877	002666	051412	DF035
1878			ERROR 140: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1879			CHECKING ZERO DETECT
1880			CS1 INCORRECT
1881	002670	057162	EM220
1882	002672	061500	EM3000
1883	002674	050704	DT035
1884	002676	051412	DF035
1885			ERROR 141: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1886			CHECKING ZERO DETECT
1887			CS2 INCORRECT
1888	002700	057162	EM220
1889	002702	063023	EM3018
1890	002704	050704	DT035
1891	002706	051412	DF035
1892			ERROR 142: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1893			CHECKING ZERO DETECT
1894			BUS ADDRESS INCORRECT
1895	002710	057162	EM220
1896	002712	063067	EM3019
1897	002714	050704	DT035
1898	002716	051412	DF035
1899			ERROR 143: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1900			CHECKING ZERO DETECT
1901			WORD COUNT INCORRECT
1902	002720	057162	EM220
1903	002722	063115	EM3020
1904	002724	050704	DT035
1905	002726	051412	DF035
1906			ERROR 144: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1907			DATA BUFFER INCORRECT
1908	002730	057102	EM219

1909	002732	063146	EM3021
1910	002734	050730	DT041
1911	002736	051436	DF041
1912	:	:	ERROR 145: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1913	:	:	CS1 INCORRECT AFTER READING DATA BUFFER
1914	002740	057102	EM219
1915	002742	063172	EM3022
1916	002744	051034	DT077
1917	002746	051612	DF077
1918	:	:	ERROR 146: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1919	:	:	CS2 INCORRECT AFTER READING DATA BUFFER
1920	002750	057102	EM219
1921	002752	063242	EM3023
1922	002754	051034	DT077
1923	002756	051612	DF077
1924	:	:	ERROR 147: UNEXPECTED MEMORY PARITY ENABLE TRAP
1925	002760	055254	EM000
1926	002762	052434	DH000C
1927	002764	050554	DT000
1928	002766	051216	DF000
1929	:	:	ERROR 150: ATTEMPTING SIMULATION OF DATA IN READ HEADER
1930	:	:	CS1 INCORRECT
1931	002770	057241	EM221
1932	002772	061500	EM3000
1933	002774	051050	DT150
1934	002776	051636	DF150
1935	:	:	ERROR 151: ATTEMPTING READ HEADER IN MAINTANENCE MODE
1936	:	:	CS1 INCORRECT AFTER COMPLETION OF COMMAND
1937	003000	057316	EM222
1938	003002	063406	EM3026
1939	003004	050740	DT042
1940	003006	051462	DF042
1941	:	:	ERROR 152: ATTEMPTING READ HEADER IN MAINTANENCE MODE
1942	:	:	CS2 INCORRECT AFTER COMPLETION OF COMMAND
1943	003010	057316	EM222
1944	003012	063455	EM3027
1945	003014	050740	DT042
1946	003016	051462	DF042
1947	:	:	ERROR 153: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
1948	:	:	CS1 INCORRECT AFTER UNLOADING DATA BUFFER
1949	003020	057371	EM223
1950	003022	063172	EM3022
1951	003024	050766	DT055
1952	003026	051532	DF055
1953	:	:	ERROR 154: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
1954	:	:	CS2 INCORRECT AFTER UNLOADING DATA
1955	003030	057371	EM223
1956	003032	063242	EM3023
1957	003034	050766	DT055
1958	003036	051532	DF055
1959	:	:	ERROR 155: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
1960	:	:	DATA READ INCORRECT
1961	003040	057371	EM223
1962	003042	063146	EM3021
1963	003044	050754	DT054
1964	003046	051506	DF054

1965			:	ERROR 156: ATTEMPTING SIMULATION OF DATA IN READ HEADER (20 BIT FORMAT)
1966			:	CS1 INCORRECT
1967	003050	057447	:	EM224
1968	003052	061500	:	EM3000
1969	003054	051050	:	DT150
1970	003056	051636	:	DF150
1971			:	ERROR 157: ATTEMPTING READ HEADER (20 BIT FORMAT) IN MAINT MODE/
1972			:	CS1 INCORRECT AFTER COMMAND COMPLETION
1973	003060	057544	:	EM225
1974	003062	063406	:	EM3026
1975	003064	050740	:	DT042
1976	003066	051462	:	DF042
1977			:	ERROR 160: ATTEMPTING READ HEADER (20 BIT FORMAT) IN MAINT MODE
1978			:	CS2 INCORRECT AFTER COMPLETION OF COMMAND
1979	003070	057544	:	EM225
1980	003072	063455	:	EM3027
1981	003074	050740	:	DT042
1982	003076	051462	:	DF042
1983			:	ERROR 161: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
1984			:	CS1 INCORRECT AFTER UNLOADING DATA BUFFER
1985	003100	057631	:	EM226
1986	003102	063172	:	EM3022
1987	003104	050766	:	DT055
1988	003106	051532	:	DF055
1989			:	ERROR 162: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
1990			:	CS2 INCORRECT AFTER UNLOADING DATA BUFFER
1991	003110	057631	:	EM226
1992	003112	063242	:	EM3023
1993	003114	050766	:	DT055
1994	003116	051532	:	DF055
1995			:	ERROR 163: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
1996			:	DATA BUFFER INCORRECT
1997	003120	057631	:	EM226
1998	003122	063146	:	EM3021
1999	003124	050754	:	DT054
2000	003126	051506	:	DF054
2001			:	ERROR 164: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2002			:	CS1 INCORRECT
2003	003130	057731	:	EM227
2004	003132	061500	:	EM3000
2005	003134	051064	:	DT164
2006	003136	051662	:	DF164
2007			:	ERROR 165: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2008			:	CS2 INCORRECT
2009	003140	057731	:	EM227
2010	003142	063023	:	EM3018
2011	003144	051064	:	DT164
2012	003146	051662	:	DF164
2013			:	ERROR 166: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2014			:	CS1 INCORRECT READING EMPTY SILO
2015	003150	057731	:	EM227
2016	003152	063524	:	EM3028
2017	003154	051064	:	DT164
2018	003156	051662	:	DF164
2019			:	ERROR 167: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2020			:	CS1 INCORRECT READING EMPTY SILO

2021	003160	057731	EM227
2022	003162	063573	EM3029
2023	003164	051064	DT164
2024	003166	051662	DF164
2025	:	:	ERROR 170: WRITE BIT ERRORS
2026	003170	000000	0
2027	003172	000000	0
2028	003174	051102	DT170
2029	003176	051706	DF170
2030	:	:	ERROR 171: WRITE GATE NOT RESET WITH SECTOR PULSE
2031	003200	060145	EM231
2032	003202	063642	EM3030
2033	003204	051126	DT171
2034	003206	051732	DF171
2035	:	:	ERROR 172: WRITE GATE NOT SET WITH SECTOR PULSE RESET
2036	003210	060267	EM232
2037	003212	063642	EM3030
2038	003214	051126	DT171
2039	003216	051732	DF171
2040	:	:	ERROR 173: WRITE GATE NOT RESET WITH SECOND INDEX PULSE
2041	003220	060515	EM235
2042	003222	063642	EM3030
2043	003224	051126	DT171
2044	003226	051732	DF171
2045	:	:	ERROR 174: CS1 INCORRECT AT END OF WRITE HEADER
2046	003230	060625	EM236
2047	003232	061500	EM3000
2048	003234	050600	DT015
2049	003236	051246	DF015
2050	:	:	ERROR 175: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2051	:	:	CS1 INCORRECT
2052	003240	061175	EM241
2053	003242	061500	EM3000
2054	003244	051136	DT175
2055	003246	051756	DF175
2056	:	:	ERROR 176: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2057	:	:	CS2 INCORRECT
2058	003250	061175	EM241
2059	003252	063023	EM3018
2060	003254	051136	DT175
2061	003256	051756	DF175
2062	:	:	ERROR 177: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2063	:	:	DRIVE STATUS REG INCORRECT
2064	003260	061175	EM241
2065	003262	063670	EM3031
2066	003264	051136	DT175
2067	003266	051756	DF175
2068	:	:	ERROR 200: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2069	:	:	ERROR REG INCORRECT
2070	003270	061175	EM241
2071	003272	063723	EM3032
2072	003274	051136	DT175
2073	003276	051756	DF175
2074	:	:	ERROR 201: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2075	:	:	CS1 INCORRECT
2076	003300	061261	EM242





```
2133 003416 052002      DF211
2134      :      ERROR 213: ATTEMPTING TO CLEAR CONTROLLER
2135      :      DRIVE STATUS REG INCORRECT
2136 003420 061446      EM244
2137 003422 063670      EM3031
2138 003424 051162      DT211
2139 003426 052002      DF211
2140      :      ERROR 214: ATTEMPTINT TO CLEAR CONTROLLER
2141      :      ERROR REG INCORRECT
2142 003430 061446      EM244
2143 003432 063723      EM3032
2144 003434 051162      DT211
2145 003436 052002      DF211
2146      .SBTTL  TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER
2147
2148 003440 000000      T.CS1: .WORD 0      ;CONTROL AND STATUS REGISTER 1
2149 003442 000000      T.WC: .WORD 0      ;WORD COUNT REGISTER
2150 003444 000000      T.BA: .WORD 0      ;BUS ADDRESS REGISTER
2151 003446 000000      T.DA: .WORD 0      ;DESIRED TRACK SECTOR REGISTER
2152 003450 000000      T.CS2: .WORD 0     ;CONTROL AND STATUS REGISTER 2
2153 003452 000000      T.DS: .WORD 0     ;DRIVE STATUS REGISTER
2154 003454 000000      T.ER: .WORD 0     ;ERROR REGISTER
2155 003456 000000      T.ASOF: .WORD 0   ;ATTENTION SUMMARY AND OFFSET REGISTER
2156 003460 000000      T.DCYL: .WORD 0  ;DESIRED CYLINDER REGISTER
2157 003462 000000      T.DB: .WORD 0     ;DATA BUFFER
2158 003464 000000      T.MR1: .WORD 0   ;MAINTENANCE REGISTER 1
2159 003466 000000      T.MR2: .WORD 0   ;MAINTENANCE REGISTER 2
2160 003470 000000      T.MR3: .WORD 0   ;MAINTENANCE REGISTER 3
2161 003472 000000      T.ECPS: .WORD 0  ;ECC POSITION INFORMATION
2162 003474 000000      T.ECPT: .WORD 0  ;ECC PATTERN INFORMATION
2163 003476 000000      T.SPARE: .WORD 0 ;SPARE REGISTER
2164
2165      .SBTTL  EXPECTED RK611 CONTROLLER REGISTERS
2166
2167 003500 000000      E.CS1: .WORD 0   ;CONTROL AND STATUS REGISTER 1
2168 003502 000000      E.WC: .WORD 0   ;WORD COUNT REGISTER
2169 003504 000000      E.BA: .WORD 0   ;BUS ADDRESS REGISTER
2170 003506 000000      E.DA: .WORD 0   ;DESIRED TRACK SECTOR REGISTER
2171 003510 000000      E.CS2: .WORD 0  ;CONTROL AND STATUS REGISTER 2
2172 003512 000000      E.DS: .WORD 0  ;DRIVE STATUS REGISTER
2173 003514 000000      E.ER: .WORD 0  ;ERROR REGISTER
2174 003516 000000      E.ASOF: .WORD 0 ;ATTENTION SUMMARY AND OFFSET REGISTER
2175 003520 000000      E.DCYL: .WORD 0 ;DESIRED CYLINDER REGISTER
2176 003522 000000      E.DB: .WORD 0  ;DATA BUFFER
2177 003524 000000      E.MR1: .WORD 0  ;MAINTENANCE REGISTER 1
2178 003526 000000      E.MR2: .WORD 0  ;MAINTENANCE REGISTER 2
2179 003530 000000      E.MR3: .WORD 0  ;MAINTENANCE REGISTER 3
2180 003532 000000      E.ECPS: .WORD 0 ;ECC POSITION INFORMATION
2181 003534 000000      E.ECPT: .WORD 0 ;ECC PATTERN INFORMATION
2182 003536 000000      E.SPARE: .WORD 0;SPARE REGISTER
2183
2184      .SBTTL  PREVIOUS RK611 CONTROLLER REGISTERS
2185
2186 003540 000000      P.CS1: .WORD 0   ;CONTROL AND STATUS REGISTER 1
2187 003542 000000      P.WC: .WORD 0   ;WORD COUNT REGISTER
2188 003544 000000      P.BA: .WORD 0   ;BUS ADDRESS REGISTER
```

```
2189 003546 000000 P.DA: .WORD 0 ;DESIRED TRACK SECTOR REGISTER
2190 003550 000000 P.CS2: .WORD 0 ;CONTROL AND STATUS REGISTER 2
2191 003552 000000 P.DS: .WORD 0 ;DRIVE STATUS REGISTER
2192 003554 000000 P.ER: .WORD 0 ;ERROR REGISTER
2193 003556 000000 P.ASOF: .WORD 0 ;ATTENTION SUMMARY AND OFFSET REGISTER
2194 003560 000000 P.DCYL: .WORD 0 ;DESIRED CYLINDER REGISTER
2195 003562 000000 P.DB: .WORD 0 ;DATA BUFFER
2196 003564 000000 P.MR1: .WORD 0 ;MAINTENANCE REGISTER 1
2197 003566 000000 P.MR2: .WORD 0 ;MAINTENANCE REGISTER 2
2198 003570 000000 P.MR3: .WORD 0 ;MAINTENANCE REGISTER 3
2199 003572 000000 P.ECPS: .WORD 0 ;ECC POSITION INFORMATION
2200 003574 000000 F.ECPT: .WORD 0 ;ECC PATTERN INFORMATION
2201 003576 000000 P.SPARE: .WORD 0 ;SPARE REGISTER
2202 .SBTTL PROGRAM DEFINED VARIABLES
2203
2204 003600 000210 RKVEC: .WORD 210 ;RK611 VECTOR
2205 003602 000240 RKPRI: .WORD PR5 ;RK611 PRIORITY
2206 003604 000000 TRAPPC: .WORD 0 ;PC FOR MEMORY CHECK ENABLE TRAP
2207 003606 000000 SRTFLG: .WORD 0 ;START FLAG
2208 : 0 = 200
2209 : 1 = 214
2210 : -1 = 204
2211 003610 000000 ERRCNT: .WORD 0 ;ERROR COUNT FOR SWITCH 12 ABORT
2212 003612 000000 P1.BIT: .WORD 0 ;NEXT BIT IN DATA SIMULATION
2213 003614 000000 PR.BIT: .WORD 0 ;PRESENT BIT IN DATA SIMULATION
2214 003616 000000 M1.BIT: .WORD 0 ;PREVIOUS BIT IN DATA SIMULATION
2215 003620 000000 M2.BIT: .WORD 0 ;BIT BEFORE PREVIOUS BIT
2216 003622 000000 BITCNT: .WORD 0 ;BIT POSITION
2217 003624 000000 WRDCNT: .WORD 0 ;WORD COUNT FOR NPR TRANSFER
2218 003626 000000 SECCNT: .WORD 0 ;SECTOR COUNT
2219 003630 000000 MEMPAR: .WORD 0 ;MEMORY ENABLE ON FIRST 24K
2220 003632 000015 WAITIM: .WORD 15 ;WAIT TIME FOR CONTROLLER READY
2221 003634 000000 SAVSWR: .WORD 0 ;STORAGE FOR SWITCH REGISTER
2222 003636 000000 PARPRE: .WORD 0 ;PARITY OPTION PRESENT FOR
2223 : LOCATION USED
2224 003640 000000 CSRPTR: .WORD 0 ;POINTER TO CSR TO BE USED
2225 .SBTTL PROGRAM SETUP
2226
2227 003642 012737 000001 003606 PARM: MOV #1,SRTFLG ;LOAD START FLAG FOR PARMETER START
2228 003650 000406 BR START1
2229
2230 003652 012737 177777 003606 RESTRT: MOV #-1,SRTFLG ;LOAD START FLAG FOR RESTART
2231 003660 000402 BR START1
2232
2233 003662 005037 003606 START: CLR SRTFLG ;CLEAR START FLAG
2234 003666 000005 START1: RESET ;RESET THE WHOLE SYSTEM
2235 003670 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
2236 003674 004737 046652 JSR PC,STKINT ;INIT KEYBOARD
2237 003700 012746 000340 MOV #PR7,-(SP) ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
2238 003704 012746 003712 MOV #1$,-(SP) ;LOAD START OF PROGRAM
2239 003710 000002 RTI ;LOAD PSW
2240
2241 003712
2242 .SBTTL INITIALIZE THE COMMON TAGS
2243 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2244 003712 012706 001100 MOV #CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
```

```

2245 003716 005026          CLR      (R6)+          ;;CLEAR MEMORY LOCATION
2246 003720 022706 001140  CMP      #SWR,R6 ;;DONE?
2247 003724 001374          BNE      -6            ;;LOOP BACK IF NO
2248 003726 012706 001100  MOV      #STACK,SP    ;;SETUP THE STACK POINTER
2249                                ;;INITIALIZE A FEW VECTORS
2250 003732 012737 044220 000020  MOV      #$$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
2251 003740 012737 000340 000022  MOV      #340,@#IOTVEC+2 ;;LEVEL 7
2252 003746 012737 045146 000030  MOV      #$$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
2253 003754 012737 000340 000032  MOV      #340,@#EMTVEC+2 ;;LEVEL 7
2254 003762 012737 050464 000034  MOV      #$$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2255 003770 012737 000340 000036  MOV      #340,@#TRAPVEC+2;LEVEL 7
2256 003776 012737 050334 000024  MOV      #$$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
2257 004004 012737 000340 000026  MOV      #340,@#PWRVEC+2 ;;LEVEL 7
2258 004012 013737 042414 042406  MOV      SENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
2259 004020 005037 001200          CLR      $TIMES        ;;INITIALIZE NUMBER OF ITERATIONS
2260 004024 005037 001202          CLR      $ESCAPE      ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2261 004030 112737 000001 001115  MOV      #1,$SERMAX    ;;ALLOW ONE ERROR PER TEST
2262 004036 012737 004036 001106  MOV      #.,$LPADR     ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2263 004044 012737 004044 001110  MOV      #.,$LPERR     ;;SETUP THE ERROR LOOP ADDRESS
2264                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2265                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2266 004052 013746 000004          MOV      @#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
2267 004056 012737 004112 000004  MOV      #64$,@#ERRVEC ;;SET UP ERROR VECTOR
2268 004064 012737 177570 001140  MOV      #DSWR,SWR     ;;SETUP FOR A HARDWARE SWICH REGISTER
2269 004072 012737 177570 001142  MOV      #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
2270 004100 022777 177777 175032  CMP      #-1,@SWR     ;;TRY TO REFERENCE HARDWARE SWR
2271 004106 001012          BNE      66$          ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2272                                ;;AND THE HARDWARE SWR IS NOT = -1
2273 004110 000403          BR      65$          ;;BRANCH IF NO TIMEOUT
2274 004112 012716 004120 64$:  MOV      #65$,(SP)    ;;SET UP FOR TRAP RETURN
2275 004116 000002          RTI
2276 004120 012737 000176 001140 65$:  MOV      #SWREG,SWR   ;;POINT TO SOFTWARE SWR
2277 004126 012737 000174 001142  MOV      #DISPREG,DISPLAY
2278 004134 012637 000004 66$:  MOV      (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
2279
2280 004140 005037 001222          CLR      $PASS        ;;CLEAR PASS COUNT
2281 004144 132737 000200 001235  BITB    #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
2282 004152 001403          BEQ     67$          ;;YES,USE NON-APT SWITCH
2283 004154 012737 001236 001140  MOV     #$$SWREG,SWR  ;;NO,USE APT SWITCH REGISTER
2284 004162
2285 004162 005037 003610 67$:  CLR     ERRCNT        ;CLEAR ERROR COUNT FOR SWITCH 12 ABORT
2286 .SBTTL TYPE PROGRAM NAME
2287 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2288 004166 005227 177777          INC     #-1          ;;FIRST TIME?
2289 004172 001047          BNE     68$          ;;BRANCH IF NO
2290 004174 022737 042550 000042  CMP     #SENDAD,@#42 ;;ACT-11?
2291 004202 001443          BEQ     68$          ;;BRANCH IF YES
2292 004204 104401 004252          TYPE   ,69$         ;;TYPE ASCIZ STRING
2293 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2294 004210 005737 000042          TST    @#42         ;;ARE WE RUNNING UNDER XXDP/ACT?
2295 004214 001012          BNE     70$          ;;BRANCH IF YES
2296 004216 123727 001234 000001  CMPB   $ENV,#1      ;;ARE WE RUNNING UNDER APT?
2297 004224 001406          BEQ     70$          ;;BRANCH IF YES
2298 004226 023727 001140 000176  CMP     SWR,#SWREG   ;;SOFTWARE SWITCH REG SELECTED?
2299 004234 001005          BNE     71$          ;;BRANCH IF NO
2300 004236 104406          GTSWR                ;;GET SOFT-SWR SETTINGS

```

```

2301 004240 000403          BR      71$
2302 004242 112737 000001 001134 70$:  MOVB   #1,$AUTOB      ;;SET AUTO-MODE INDICATOR
2303 004250          71$:
2304 004250 000420          BR      68$          ;;GET OVER THE ASCIZ
2305          ;;69$: .ASCIZ <CRLF>/CZR6CDO RK611 DSKLS CTRL PRT3/<CRLF>
2306 004312          68$:
2307 004312 005227 177777          INC    #-1          ;TEST IF FIRST PASS
2308 004316 001002          BNE    6$          ;NO - SKIP
2309 004320 104401 052150          TYPE  ,OPR006      ;TYPE RUN TIME MESSAGE
2310 004324 022737 000001 003606 6$:  CMP    #1,SRTFLG   ;CHECK IF PARAMETER START
2311 004332 001122          BNE    15$        ;NO, CONTINUE SETUP
2312 004334 104401 052036          TYPE  ,OPR001      ;TYPE 'RK611 BUS ADDRESS ( ) ='
2313 004340 013746 001270          MOV    $BASE,-(SP) ;SAVE $BASE FOR TYPEOUT
2314 004344 104402          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2315 004346 104401 052070          TYPE  ,OPR002
2316 004352 104412          RDOCT          ;GET VALUE
2317 004354 012637 001160          MOV    (SP)+,$TMPO
2318 004360 001407          BEQ    7$          ;CHECK IF <CR>
2319 004362 022737 160000 001160          CMP    #160000,$TMPO ;CHECK IF IN I/O PAGE
2320 004370 101361          BHI    5$
2321 004372 013737 001160 001270          MOV    $TMPO,$BASE ;LOAD NEW BUS ADDRESS
2322 004400 104401 052076          TYPE  ,OPR003      ;TYPE 'RK611 VECTOR ADDRESS ( ) ='
2323 004404 013746 001264          MOV    $VECT1,-(SP) ;TYPE OUT VECTOR ADDRESS
2324 004410 042716 160000          BIC    #160000,(SP)
2325 004414 104402          TYPOC
2326 004416 104401 052070          TYPE  ,OPR002
2327 004422 104412          RDOCT          ;GET VALUE
2328 004424 012637 001160          MOV    (SP)+,$TMPO
2329 004430 001412          BEQ    10$         ;CHECK IF <CR>
2330 004432 022737 001000 001160          CMP    #1000,$TMPO  ;CHECK IF LEGAL
2331 004440 101757          BLOS   7$
2332 004442 042737 017777 001264          BIC    #17777,$VECT1 ;LOAD NEW VECTOR ADDRESS
2333 004450 053737 001160 001264          BIS    $TMPO,$VECT1
2334 004456 104401 052126          10$:  TYPE  ,OPR004      ;TYPE 'RK611 PRIORITY ( ) ='
2335 004462 005046          CLR    -(SP)
2336 004464 113716 001265          MOVB   $VECT1+1,(SP)
2337 004470 006216          ASR    (SP)        ;SHIFT 5 BITS RIGHT
2338 004472 006216          ASR    (SP)
2339 004474 006216          ASR    (SP)
2340 004476 006216          ASR    (SP)
2341 004500 006216          ASR    (SP)
2342 004502 104402          TYPOC
2343 004504 104401 052070          TYPE  ,OPR002
2344 004510 104412          RDOCT          ;GET VALUE
2345 004512 012637 001160          MOV    (SP)+,$TMPO
2346 004516 001430          BEQ    15$         ;CHECK FOR DEFAULT
2347 004520 022737 000007 001160          CMP    #7,$TMPO    ;CHECK IF LEGAL
2348 004526 103753          BLO    10$
2349 004530 022737 000004 001160          CMP    #4,$TMPO
2350 004536 101347          BHI    10$
2351 004540 006337 001160          ASL    $TMPO      ;SHIFT 5 BITS LEFT
2352 004544 006337 001160          ASL    $TMPO
2353 004550 006337 001160          ASL    $TMPO
2354 004554 006337 001160          ASL    $TMPO
2355 004560 006337 001160          ASL    $TMPO
2356 004564 042737 160000 001264          BIC    #160000,$VECT1 ;STORE NEW PRIORITY

```

CZR6CDO RK611 DSKLS CTRL PRT3  
CZR6CD.P11 19-JUN-80 13:49

MACY11 30A(1052) 19-JUN-80 13:52 PAGE 46  
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0045

```
2357 004572 153737 001160 001265      BISB    $TMP0,$VECT1+1
2358 004600 013737 001264 003600 15$:    MOV     $VECT1,RKVEC      ;STORE RK611 VECTOR
2359 004606 042737 160000 003600      BIC     #160000,RKVEC
2360 004614 113737 001265 003602      MOVB   $VECT1+1,RKPRI  ;STORE PRIORITY
2361 004622 004737 043712      JSR    PC,$SIZE        ;SIZE MEMORY
2362 004626 013702 001270      MOV     $BASE,R2       ;SET RK611 BASE
2363 004632 005037 001202      CLR    $ESCAPE        ;CLEAR ESCAPE
2364
2365 004636 012746 000000      NEWPAS: MOV   #PRO,-(SP) ;ALLOW ALL INTERRUPTS
2366 004642 012746 004650      MOV    #TST1,-(SP)
2367 004646 000002      RTI
```

2368  
2369  
2370  
2371  
2372  
2373  
2374  
2375  
2376  
2377  
2378  
2379  
2380  
2381  
2382  
2383  
2384  
2385  
2386  
2387  
2388  
2389  
2390  
2391  
2392  
2393  
2394  
2395  
2396  
2397  
2398  
2399  
2400  
2401  
2402  
2403  
2404  
2405  
2406  
2407  
2408  
2409  
2410  
2411  
2412  
2413  
2414  
2415  
2416  
2417  
2418  
2419  
2420  
2421  
2422  
2423

.SBTTL \*\*DRIVE MESSAGES FOR CLASS B INSTRUCTIONS

\*\*\*\*\*  
\*TEST 1 READ HEADER SEEK MESSAGE

\*  
\* CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.  
\* PUT THE RK611 CONTROLLER IN DIAGNOSTIC MODE. ISSUE  
\* A READ HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0.  
\* HEAD 0, DRIVE 0. CLOCK IN SEEK MESSAGE INTO SHIFT REGISTER.  
\* VERIFY THAT A SEEK IS LOADED WITH THE PROPER BITS IN  
\* MESSAGE SET. REPEAT FOR A READ HEADER WITH CDT SET  
\* IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.  
\*

\*\*\*\*\*  
TST1: SCOPE  
MOV #100.,\$TIMES ;;DO 100. ITERATIONS  
MOV \$BASE,R2 ;LOAD RK611 BASE  
MOV #CCLR,RKCS1(R2) ;CLEAR RK611  
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE  
MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.  
MOV #3400,RKDA(R2) ;LOAD TRACK  
MOV #7,RKCS2(R2) ;LOAD DRIVE NUM.  
MOV #CDT!CFMT!RDHEAD,RKCS1(R2) ;ISSUE RDHEAD WITH CDT SET IN  
; 24 SECTOR FORMAT  
;CLOCK IN DRIVE MESSAGE  
1\$: MOV #3\*4+2,R0  
MOV #DMD!MCLK,RKMR1(R2)  
MOV #DMD,RKMR1(R2)  
DEC R0  
BNE 1\$  
MOV RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1  
MOV RKMR2(R2),T.MR2 ;STORE MAINT REG. 2 (MESS A)  
MOV RKMR3(R2),T.MR3 ;STORE MAINT REG. 3 (MESS B)  
MOV #CDT!CFMT!RDHEAD,E.CS1 ;LOAD EXPECTED CS1  
MOV #S.SEEK!S.FMT!70007,E.MR2 ;LOAD EXPECTED MR2  
MOV #37760,E.MR3 ;LOAD EXPECTED MR3  
CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT  
BEQ 2\$ ;YES, CHECK MESSAGE A  
ERROR 1 ;CS1 INCORRECT  
MOV #CCLR,RKCS1(R2) ;CLEAR RK611  
BR TST2 ;GO TO NEXT TEST  
2\$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT  
BEQ 3\$ ;YES, CHECK MESSAGE B  
ERROR 2 ;MESS A INCORRECT  
3\$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT  
BEQ TST2 ;YES, GO ON TO NEXT TEST  
ERROR 3 ;MESS B INCORRECT

\*\*\*\*\*  
\*TEST 2 WRITE HEADER SEEK MESSAGE

\*  
\* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
\* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06  
\* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
\* CLOCK IN SEEK MESSAGE INTO SHIFT REGISTER. VERIFY  
\* THAT A SEEK IS LOADED WITH THE RTC BIT SET. REPEAT  
\* FOR A WRITE HEADER WITH CDT SET IN 24 SECTOR FORMAT.  
\*

```
2424
2425
2426
2427 005066 000004
2428 005070 012737 000144 001200
2429 005076 013702 001270
2430 005102 012762 100000 000000
2431 005110 012762 000040 000026
2432 005116 012762 001777 000020
2433 005124 012762 003400 000006
2434 005132 012762 000007 000010
2435 005140 012762 012027 000000
2436
2437 005146 012700 000016
2438 005152 012762 000440 000026 1$:
2439 005160 012762 000040 000026
2440 005166 005300
2441 005170 001370
2442 005172 016237 000000 003440
2443 005200 016237 000034 003466
2444 005206 016237 000036 003470
2445 005214 012737 012027 003500
2446 005222 012737 071227 003526
2447 005230 012737 037760 003530
2448 005236 023737 003500 003440
2449 005244 001405
2450 005246 104004
2451 005250 012762 100000 000000
2452 005256 000412
2453 005260 023737 003526 003466 2$:
2454 005266 001401
2455 005270 104005
2456 005272 023737 003530 003470 3$:
2457 005300 001401
2458 005302 104006
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470 005304 000004
2471 005306 012737 000144 001200
2472 005314 013702 001270
2473 005320 012762 100000 000000
2474 005326 012762 000040 000026
2475 005334 012762 001777 000020
2476 005342 012762 003400 000006
2477 005350 012762 000007 000010
2478 005356 012762 012025 000000
2479
```

```
*****
CYLINDER 1777, HEAD 7, DRIVE 7.
*****
TST2: SCOPE
MOV #100.,$TIMES ;;DO 100. ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
MOV #3400,RKDA(R2) ;LOAD TRACK
MOV #7,RKCS2(R2) ;LOAD DRIVE NUM.
MOV #CDT!CFMT!WRHEAD,RKCS1(R2) ;ISSUE WRHEAD WITH CDT SET IN
; 24 SECTOR FORMAT
;CLOCK IN DRIVE MESSAGE
MOV #3*4+2,R0
MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1
MOV RKMR2(R2),T.MR2 ;STORE MAINT REG. 2 (MESS A)
MOV RKMR3(R2),T.MR3 ;STORE MAINT REG. 3 (MESS B)
MOV #CDT!CFMT!WRHEAD,E.CS1 ;LOAD EXPECTED CS1
MOV #S.SEEK!S.RTC!S.FMT!70007,E.MR2 ;LOAD EXPECTED MR2
MOV #37760,E.MR3 ;LOAD EXPECTED MR3
CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
BEQ 2$ ;YES, CHECK MESSAGE A
ERROR 4 ;CS1 INCORRECT
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
BR TST3 ;GO TO NEXT TEST
CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
BEQ 3$ ;YES, CHECK MESSAGE B
ERROR 5 ;MESS A INCORRECT
CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
BEQ TST3 ;YES, GO ON TO NEXT TEST
ERROR 6 ;MESS B INCORRECT
*****
TEST 3 READ HEADER DRIVE CLEAR MESSAGE
*****
CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
IN DIAGNOSTIC MODE. ISSUE A READ HEADER WITH CDT SET
IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.
CLOCK SEEK MESSAGE AND MAKE SURE A DRIVE CLEAR IS
GENERATED AND THE PROPER BITS ARE SET.
*****
TST3: SCOPE
MOV #100.,$TIMES ;;DO 100. ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
MOV #3400,RKDA(R2) ;LOAD TRACK
MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER
MOV #CDT!CFMT!RDHEAD,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
; 24 SECTOR FORMAT
```



```

2480 005364 012700 000156          MOV      #27.*4+2,R0      ;LOAD COUNT TO LOAD DRIVE CLEAR
2481 005370 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2)
2482 005376 012762 000040 000026  MOV      #DMD,RKMR1(R2)
2483 005404 005300          DEC      R0
2484 005406 001370          BNE     1$
2485 005410 016237 000000 003440  MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2486 005416 016237 000034 003466  MOV      RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2487 005424 016237 000036 003470  MOV      RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2488 005432 012737 012025 003500  MOV      #CDT!CFMT!RDHEAD,E.CS1 ;LOAD EXPECTED CS1
2489 005440 012737 071407 003526  MOV      #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2490 005446 005037 003530          CLR     E.MR3           ;LOAD EXPECTED MAINT REG.
2491 005452 023737 003500 003440  CMP      E.CS1,T.CS1    ;CHECK COMMAND AND STATUS REG 1 CORRECT
2492 005460 001405          BEQ     2$             ;YES, CHECK CS2
2493 005462 104007          ERROR   7              ;CS1 INCORRECT
2494 005464 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2495 005472 000412          BR      TST4           ;GO TO NEXT TEST
2496 005474 023737 003526 003466 2$:  CMP      E.MR2,T.MR2    ;CHECK MESS A CORRECT
2497 005502 001401          BEQ     3$             ;YES, CHECK MESS B
2498 005504 104010          ERROR   10            ;MESS A INCORRECT
2499 005506 023737 003530 003470 3$:  CMP      E.MR3,T.MR3    ;CHECK MESS B CORRECT
2500 005514 001401          BEQ     TST4           ;YES, GO ON TO NEXT TEST
2501 005516 104011          ERROR   11            ;MESS B INCORRECT
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513

```

```

*****
*TEST 4      WRITE HEADER DRIVE CLEAR MESSAGE
*****
*
*   CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
*   IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER WITH CDT SET
*   IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.
*   CLOCK SEEK MESSAGE AND LOAD GENERATED DRIVE CLEAR
*   INTO SHIFT REGISTER.  MAKE SURE THE DRIVE CLEAR IS
*   GENERATED AND THE PROPER BITS ARE SET.
*****

```

```

2514 005520 000004          TST4:  SCOPE
2515 005522 012737 000144 001200  MOV      #100.,$TIMES    ;;DO 100. ITERATIONS
2516 005530 013702 001270          MOV      $BASE,R2       ;LOAD RK611 BASE
2517 005534 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2518 005542 012762 000040 000026  MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2519 005550 012762 001777 000020  MOV      #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2520 005556 012762 003400 000006  MOV      #3400,RKDA(R2)  ;LOAD TRACK
2521 005564 012762 000007 000010  MOV      #7,RKCS2(R2)    ;LOAD DRIVE NUMBER
2522 005572 012762 012027 000000  MOV      #CDT!CFMT!WRHEAD,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2523                                     ; 24 SECTOR FORMAT
2524 005600 012700 000156          MOV      #27.*4+2,R0    ;LOAD COUNT TO LOAD DRIVE CLEAR
2525 005604 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2)
2526 005612 012762 000040 000026  MOV      #DMD,RKMR1(R2)
2527 005620 005300          DEC      R0
2528 005622 001370          BNE     1$
2529 005624 016237 000000 003440  MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2530 005632 016237 000034 003466  MOV      RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2531 005640 016237 000036 003470  MOV      RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2532 005646 012737 012027 003500  MOV      #CDT!CFMT!WRHEAD,E.CS1 ;LOAD EXPECTED CS1
2533 005654 012737 071407 003526  MOV      #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2534 005662 005037 003530          CLR     E.MR3           ;LOAD EXPECTED MAINT REG.
2535 005666 023737 003500 003440  CMP      E.CS1,T.CS1    ;CHECK COMMAND AND STATUS REG 1 CORRECT

```

```

2536 005674 001405 BEQ 2$ ;YES, CHECK CS2
2537 005676 104012 ERROR 12 ;CS1 INCORRECT
2538 005700 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2539 005706 000412 BR TST5 ;GO TO NEXT TEST
2540 005710 023737 003526 003466 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2541 005716 001401 BEQ 3$ ;YES, CHECK MESS B
2542 005720 104013 ERROR 13 ;MESS A INCORRECT
2543 005722 023737 003530 003470 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2544 005730 001401 BEQ TST5 ;YES, GO ON TO NEXT TEST
2545 005732 104014 ERROR 14 ;MESS B INCORRECT
    
```

.SBTTL \*\*INDEX AND SECTOR PULSE DETECT ON

```

*****
*TEST 5 SECTOR PULSE DETECT IN READ HEADER (PART 1)
*
* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
* IN 26 SECTOR MODE, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK ANY DRIVE CLEAR MESSAGES.
* SIMULATE SECTOR PULSE, 255 ZEROES AND A ONE.
*
* MAKE SURE READ GATE DOES SET.
    
```

```

*****
2561 TST5: SCOPE
2562 005734 000004 MOV #10,$TIMES ;DO 10. ITERATIONS
2563 005736 012737 000012 001200 MOV $BASE,R2 ;LOAD RK611 BASE
2564 005744 013702 001270 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2565 005750 012762 100000 000000 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2566 005756 012762 000040 000026 MOV #DMD!MSP,RKMR1(R2) ;INITIALIZE ROM ADDRESS
2567 005764 012762 000140 000026 MOV #DMD,RKMR1(R2)
2568 005772 012762 000040 000026 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
2569 006000 012762 000025 000000 MOV #50.*4+2,R0 ;CLOCK UNTIL READY FOR SECTOR PULSE
2570 006006 012700 000312 1$: MOV #DMD!MCLK,RKMR1(R2)
2571 006012 012762 000440 000026 MOV #DMD,RKMR1(R2)
2572 006020 012762 000040 000026 DEC R0
2573 006026 005300 BNE 1$
2574 006030 001370 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2575 006032 016237 000000 003440 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
2576 006040 012737 000025 003500 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2577 006046 023737 003500 003440 BEQ 2$ ;YES, CLOCK ZEROES
2578 006054 001405 ERROR 15 ;CS1 INCORRECT
2579 006056 104015 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2580 006060 012762 100000 000000 BR TST6 ;GO ON TO NEXT TEST
2581 006066 000553
2582
2583 006070 012762 000140 000026 2$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
2584 006076 012762 000040 000026 MOV #DMD,RKMR1(R2)
2585 006104 012737 022040 003524 MOV #DMD!NEWD!ECCW,E.MR1 ;LOAD EXPECT MAINT REG. 1
2586 006112 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
2587 006116 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT AND PREVIOUS
2588 006122 005037 003616 CLR M1.BIT ;BITS TO GENERATE ZEROES
2589 006126 012700 000200 MOV #128,R0 ;GENERATE 128 ZEROS UNTIL READ GATE
2590 006132 004737 043540 5$: JSR PC,RDBIT ;READ A ZERO
2591 006136 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINT REG. 1
    
```

```

2592 006144 023737 003524 003464    CMP    E.MR1,T.MR1    ;CHECK MAINTENANCE REG. 1 CORRECT
2593 006152 001405                BEQ    6$            ;YES, SIMULATE NEXT BIT
2594 006154 104017                ERROR  17           ;MAINT REG. 1 INCORRECT
2595 006156 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
2596 006164 000514                BR     TST6         ;;GO ON TO NEXT TEST
2597
2598 006166 005237 003622    6$:   INC    BITCNT        ;INCREMENT BIT COUNT
2599 006172 005300                DEC    R0           ;CHECK READY OF READ GATE
2600 006174 001356                BNE   5$            ;NO, CONTINUE
2601 006176 012737 122040 003524    MOV    #DMD!MEWD!ECCW!RDGATE,E.MR1 ;LOAD EXPECTED MR1
2602 006204 012700 000177    MOV    #127,R0      ;GENERATE 127 ZEROS
2603 006210 004737 043540    17$:  JSR    PC,RDBIT     ;READ A ZERO
2604 006214 016237 000026 003464    MOV    RKMR1(R2),T.MR1 ;STORE MAINT REG. 1
2605 006222 023737 003524 003464    CMP    E.MR1,T.MR1   ;CHECK MAINT REG. 1 CORRECT
2606 006230 001405                BEQ    11$         ;YES, SIMULATE NEXT BIT
2607 006232 104017                ERROR  17           ;MAINT REG. 1 INCORRECT
2608 006234 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
2609 006242 000465                BR     TST6         ;;GO ON TO NEXT TEST
2610
2611 006244 005237 003622    11$:  INC    BITCNT        ;INCREMENT BIT COUNT
2612 006250 005300                DEC    R0           ;CHECK IF ALL ZEROS ISSUED
2613 006252 001356                BNE   10$          ;NO, CONTINUE
2614 006254 012737 000001 003614    MOV    #1,PR.BIT    ;LOAD ONE FOR READING 1
2615 006262 004737 043540    JSR    PC,RDBIT     ;READ A ONE
2616 006266 016237 000026 003464    MOV    RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG.
2617 006274 023737 003524 003464    CMP    E.MR1,T.MR1   ;CHECK MAINT REG. 1
2618 006302 001405                BEQ    12$         ;YES, CONTINUE
2619 006304 104017                ERROR  17           ;MAINTENANCE REG. 1 INCORRECT
2620 006306 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
2621 006314 000440                BR     TST6         ;;GO ON TO NEXT TEST
2622 006316 005237 003622    12$:  INC    BITCNT        ;INCREMENT BIT COUNT
2623 006322 013737 003614 003616    MOV    PR.BIT,M1.BIT ;LOAD ZERO FOR NEXT BIT
2624 006330 005037 003614    CLR    PR.BIT
2625 006334 004737 043540    JSR    PC,RDBIT     ;SIMULATE ZERO
2626 006340 016237 000026 003464    MOV    RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2627 006346 023737 003524 003464    CMP    E.MR1,T.MR1   ;CHECK MAINT REG. 1 CORRECT
2628 006354 001405                BEQ    13$         ;CHECK CS1 CORRECT
2629 006356 104017                ERROR  17           ;MAINTENANCE REG. 1 INCORRECT
2630 006360 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
2631 006366 000413                BR     TST6         ;;GO TO NEXT TEST
2632
2633 006370 016237 000000 003440 13$:  MOV    RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2634 006376 012737 000025 003500    MOV    #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
2635 006404 023737 003500 003440    CMP    E.CS1,T.CS1   ;CHECK CS1 CORRECT
2636 006412 001401                BEQ    TST6        ;YES, GO TO NEXT TEST
2637 006414 104016                ERROR  16           ;CS1 INCORRECT

```

```

2638
2639 .....
2640 *TEST 6          SECTOR PULSE DETECT IN READ HEADER (PART 2)
2641 *
2642 *
2643 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
2644 * IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
2645 * IN 26 SECTOR MODE, CYLINDER 0, HEAD 0, DRIVE 0.
2646 * CLOCK BOTH SEEK ANY DRIVE CLEAR MESSAGES.
2647 * SIMULATE INDEX PULSE, 255 ZEROES AND A ONE.

```

```
2648      : * MAKE SURE READ GATE DOES NOT SET.
2649      : *
2650      : * *****
2651 006416 000004          TST6: SCOPE
2652 006420 012737 000012 001200  MOV #10,STIMES      ;;DO 10. ITERATIONS
2653 006426 013702 001270          MOV SBASE,R2        ;;LOAD RK611 BASE
2654 006432 012762 100000 000000  MOV #CCLR,RKCS1(R2) ;;CLEAR RK611
2655 006440 012762 000040 000000  MOV #DMD,RKCS1(R2)  ;;PUT RK611 IN DIAGNOSTIC MODE
2656 006446 012762 000140 000026  MOV #DMD!MSP,RKMR1(R2) ;;INITIALIZE ROM ADDRESS
2657 006454 012762 000040 000026  MOV #DMD,RKMR1(R2)
2658 006462 012762 000025 000000  MOV #RDHEAD,RKCS1(R2) ;;ISSUE READ HEADER
2659 006470 012700 000312          MOV #50,*4+2,R0    ;;CLOCK UNTIL READY FOR SECTOR PULSE
2660 006474 012762 000440 000026 1S: MOV #DMD!MCLK,RKMR1(R2)
2661 006502 012762 000040 000026  MOV #DMD,RKMR1(R2)
2662 006510 005300          DEC R0
2663 006512 001370          BNE 1S
2664 006514 016237 000000 003440  MOV RKCS1(R2),T.CS1 ;;STORE COMMAND AND STATUS REG. 1
2665 006522 012737 000025 003500  MOV #RDHEAD,E.CS1  ;;LOAD EXPECTED CS1
2666 006530 023737 003500 003440  CMP E.CS1,T.CS1    ;;CHECK COMMAND AND STATUS REG. 1 CORRECT
2667 006536 001405          BEQ 2S             ;;YES, CLOCK IN ZEROS
2668 006540 104015          ERROR 1S
2669 006542 012762 100000 000000  MOV #CCLR,RKCS1(R2) ;;CLEAR RK611
2670 006550 000535          BR TST7           ;;GO ON TO NEXT TEST
2671 006552          2S:
2672 006552 012762 000240 000026  MOV #DMD!MIND,RKMR1(R2) ;;SIMULATE INDX PULSE
2673 006560 012700 000004          MOV #4,R0
2674 006564 012762 000640 000026 3S: MOV #DMD!MIND!MCLK,RKMR1(R2)
2675 006572 012762 000240 000026  MOV #DMD!MIND,RKMR1(R2)
2676 006600 005300          DEC R0
2677 006602 001370          BNE 3S
2678 006604 012762 000040 000026  MOV #DMD,RKMR1(R2)
2679 006612 005037 003622          CLR BITCNT        ;;INITIALIZE BIT COUNT
2680 006616 012737 022040 003524  MOV #DMD!MEWD!ECCW,E.MR1 ;;LOAD EXPECTED MAINTENANCE REG. 1
2681 006624 005037 003614          CLR PR.BIT        ;;INITIALIZE PRESENT AND PREVIOUS
2682 006630 005037 003616          CLR M1.BIT        ;;BITS TO GENERATE ZEROS
2683 006634 012700 000377          MOV #255.,R0      ;;GENERATE 255 ZEROES
2684 006640 004737 043540          JSR PC,RDBIT      ;;READ A ZERO
2685 006644 016237 000026 003464 5S: MOV RKMR1(R2),T.MR1 ;;STORE MAINTENANCE REG. 1
2686 006652 023737 003524 003464  CMP E.MR1,T.MR1   ;;CHECK READ GATE NOT SET
2687 006660 001405          BEQ 6S           ;;READ GATE NOT SET SIMULATE NEXT BIT
2688 006662 104020          ERROR 20        ;;MAINT REG. 1 INCORRECT
2689 006664 012762 100000 000000  MOV #CCLR,RKCS1(R2) ;;ISSUE CONTROLLER CLEAR
2690 006672 000464          BR TST7           ;;GO ON TO NEXT TEST
2691
2692 006674 005237 003622          6S: INC BITCNT        ;;INCREMENT BIT COUNT
2693 006700 005300          DEC R0            ;;CHECK IF ALL ZEROES ISSUED
2694 006702 001356          BNE 5S           ;;NO, CONTINUE
2695 006704 012737 000001 003614  MOV #1,PR.BIT     ;;LOAD ONE FOR READING 1
2696 006712 004737 043540          JSR PC,RDBIT     ;;READ A ONE
2697 006716 016237 000026 003464  MOV RKMR1(R2),T.MR1 ;;STORE MAINTENANCE REG. 1
2698 006724 023737 003524 003464  CMP E.MR1,T.MR1   ;;CHECK READ GATE NOT SET
2699 006732 001405          BEQ 7S           ;;YES, CONTINUE
2700 006734 104020          ERROR 20        ;;MAINT REG. 1 INCORRECT
2701 006736 012762 100000 000000  MOV #CCLR,RKCS1(R2) ;;ISSUE CONTROLLER CLEAR
2702 006744 000437          BR TST7           ;;GO ON TO NEXT TEST
2703
```

```
2704 006746 005237 003622 78: INC BITCNT ;INCREMENT BIT COUNT
2705 006752 013737 003614 003616 MOV PR.BIT,M1.BIT ;LOAD ZERO FOR NEXT BIT
2706 006760 005037 003614 CLR PR.BIT
2707 006764 004737 043540 JSR PC,RDBIT ;SIMULATE ZERO
2708 006770 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2709 006776 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MAINTENANCE REG. 1 CORRECT
2710 007004 001404 BEQ 98 ;CHECK CS1 CORRECT
2711 007006 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2712 007014 000413 BR TST7 ;;GO TO NEXT TEST
2713
2714 007016 016237 000000 003440 98: MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2715 007024 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
2716 007032 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
2717 007040 001401 BEQ TST7 ;;YES, GO TO NEXT TEST
2718 007042 104016 ERROR 16 ;CS1 INCORRECT
2719
```

\*\*\*\*\*  
:TEST 7 SECTOR PULSE DETECT IN READ HEADER (PART 3)  
\*\*\*\*\*

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06  
IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
CLOCK BOTH SEEK AND DRIVE CHECK MESSAGES.  
SIMULATE 255 ZEROES AND A ONE.

MAKE SURE READ GATE DOES NOT SET.

```
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732 007044 000004 TST7: SCOPE
2733 007046 012737 000012 001200 MOV #10,,STIMES ;;DO 10. ITERATIONS
2734 007054 013702 001270 MOV SBASE,R2 ;LOAD RK611 BASE
2735 007060 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2736 007066 012762 000040 000000 MOV #DMD,RKCS1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2737 007074 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;INITIALIZE ROM ADDRESS
2738 007102 012762 000040 000026 MOV #DMD,RKMR1(R2)
2739 007110 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
2740 007116 012700 000312 MOV #50,+4+2,RO ;CLOCK UNTIL READY FOR SECTOR PULSE
2741 007122 012762 000440 000026 18: MOV #DMD!MCLK,RKMR1(R2)
2742 007130 012762 000040 000026 MOV #DMD,RKMR1(R2)
2743 007136 005300 DEC RO
2744 007140 001370 BNE 18
2745 007142 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2746 007150 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
2747 007156 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2748 007164 001405 BEQ 28 ;YES, CLOCK IN ZEROS
2749 007166 104015 ERROR 15
2750 007170 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2751 007176 000515 BR TST10 ;;GO ON TO NEXT TEST
2752
2753 007200 005037 003622 28: CLR BITCNT ;INITIALIZE BIT COUNT
2754 007204 012737 022040 003524 MOV #DMD!MEND!ECCW,E.MR1 ;LOAD EXPECTED MAINTENANCE REG. 1
2755 007212 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT AND PREVIOUS
2756 007216 005037 003616 CLR M1.BIT ;BITS TO GENERATE ZEROS
2757 007222 012700 000377 MOV #255,,RO ;GENERATE 255 ZEROES
2758 007226 004737 043540 58: JSR PC,RDBIT ;READ A ZERO
2759 007232 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
```

```

2760 007240 023737 003524 003464    CMP    E.MR1,T.MR1    ;CHECK READ GATE NOT SET
2761 007246 001405                BEQ    6$             ;READ GATE NOT SET SIMULATE NEXT BIT
2762 007250 104021                ERROR  21            ;MAINT REG. 1 INCORRECT
2763 007252 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;ISSUE CONTROLLER CLEAR
2764 007260 000464                BR     TST10         ;;GO ON TO NEXT TEST
2765
2766 007262 005237 003622    6$:   INC    BITCNT      ;INCREMENT BIT COUNT
2767 007266 005300                DEC    R0             ;CHECK IF ALL ZEROES ISSUED
2768 007270 001356                BNE   5$             ;NO, CONTINUE
2769 007272 012737 000001 003614    MOV    #1,PR.BIT     ;LOAD ONE FOR READING 1
2770 007300 004737 043540                JSR   PC,RDBIT       ;READ A LINE
2771 007304 016237 000026 003464    MOV    RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2772 007312 023737 003524 003464    CMP    E.MR1,T.MR1   ;CHECK READ GATE NOT SET
2773 007320 001405                BEQ    7$             ;YES, CONTINUE
2774 007322 104021                ERROR  21            ;MAINT REG. 1 INCORRECT
2775 007324 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;ISSUE CONTROLLER CLEAR
2776 007332 000437                BR     TST10         ;;GO ON TO NEXT TEST
2777
2778 007334 005237 003622    7$:   INC    BITCNT      ;INCREMENT BIT COUNT
2779 007340 013737 003614 003616    MOV    PR.BIT,M1.BIT ;LOAD ZERO FOR NEXT BIT
2780 007346 005037 003614                CLR   PR.BIT
2781 007352 004737 043540                JSR   PC,RDBIT       ;SIMULATE ZERO
2782 007356 016237 000026 003464    MOV    RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2783 007364 023737 003524 003464    CMP    E.MR1,T.MR1   ;CHECK MAINTENANCE REG. 1 CORRECT
2784 007372 001404                BEQ    9$             ;CHECK CS1 CORRECT
2785 007374 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
2786 007402 000413                BR     TST10         ;;GO TO NEXT TEST
2787
2788 007404 016237 000000 003440  9$:   MOV    RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2789 007412 012737 000025 003500    MOV    #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
2790 007420 023737 003500 003440    CMP    E.CS1,T.CS1   ;CHECK CS1 CORRECT
2791 007426 001401                BEQ    TST10         ;;YES, GO TO NEXT TEST
2792 007430 104016                ERROR  16            ;CS1 INCORRECT

```

```

*****
:TEST 10      INDEX PULSE DETECTION IN WRITE HEADER
*****

```

```

:
: CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT
: THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER
: TO AN RK06, IN 26 SECTOR FORMAT, TO CYLINDER 0, HEAD 0,
: DRIVE 0, WITH A ONE WORD TRANSFER. CLOCK THROUGH THE
: SEEK AND THE DRIVE CLEAR MESSAGES. ISSUE 200 CONTROLLER
: CLOCKS AND MAKE SURE WRITE GATE DOES NOT SET. SIMULATE
: SECTOR PULSE AND 200 CONTROLLER CLOCKS MAKING SURE WRITE
: GATE DOES NOT SET. SIMULATE INDEX PULSE AND MAKE SURE
: WRITE GATE SETS.

```

```

2807
2808 007432 000004                TST10: SCOPE
2809 007434 012737 000012 001200    MOV    #10,$TIMES    ;;DO 10. ITERATIONS
2810 007442 013702 001270                MOV    $BASE,R2      ;LOAD RK611 BASE
2811 007446 012762 100000 000000    MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
2812 007454 012762 000040 000026    MOV    #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2813 007462 012762 000140 000026    MOV    #DMD!MSP,RKMR1(R2) ;INITIALIZE ROM ADDRESS
2814 007470 012762 000040 000026    MOV    #DMD,RKMR1(R2)
2815 007476 012762 064630 000004    MOV    #WRBUFF,RKBA(R2) ;LOAD BUS ADDRESS

```

CZR6CDO RK611 DSKLS CTRL PRT3  
CZR6CD.P11 19-JUN-80 13:49

MACY11 30A(1052) 19-JUN-80 13:52 PAGE 55  
T10 INDEX PULSE DETECTION IN WRITE HEADER

SEQ 0054

```
2816 007504 012762 177777 000002      MOV      #-1,RKWC(R2)      ;LOAD WORD COUNT
2817 007512 012762 000027 000000      MOV      #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
2818 007520 012700 000312          MOV      #50,+4+2,RO      ;CLOCK UNTIL READY FOR INDEX PULSE
2819 007524 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2)
2820 007532 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2821 007540 005300          DEC      RO
2822 007542 001370          BNE     1$
2823 007544 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2824 007552 016237 000004 003444      MOV      RKBA(R2),T.BA   ;STORE BUS ADDRESS REG.
2825 007560 016237 000002 003442      MOV      RKWC(R2),T.WC   ;STORE WORD COUNT REG.
2826 007566 012737 000027 003500      MOV      #WRHEAD,E.CS1  ;LOAD EXPECTED CS1
2827 007574 012737 064630 003504      MOV      #WRBUFF,E.BA   ;LOAD EXPECTED BUS ADDRESS
2828 007602 012737 177777 003502      MOV      #-1,E.WC       ;LOAD EXPECTED WORD COUNT
2829 007610 023737 003500 003440      CMP      E.CS1,T.CS1    ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2830 007616 001403          BEQ     2$              ;YES, CONTINUE
2831 007620 104022          ERROR  22              ;CS1 INCORRECT
2832 007622 000137 010550          JMP     60$            ;CLEAR CONTROLLER
2833
2834 007626 023737 003504 003444 2$:      CMP      E.BA,T.BA      ;CHECK BUS ADDRESS CORRECT
2835 007634 001403          BEQ     3$              ;YES, CONTINUE
2836 007636 104023          ERROR  23              ;BUS ADDRESS INCORRECT
2837 007640 000137 010550          JMP     60$            ;CLEAR RK611
2838
2839 007644 023737 003502 003442 3$:      CMP      E.WC,T.WC      ;CHECK WORD COUNT CORRECT
2840 007652 001403          BEQ     4$              ;YES, CONTINUE
2841 007654 104024          ERROR  24              ;WORD COUNT INCORRECT
2842 007656 000137 010550          JMP     60$            ;CLEAR RK611
2843
2844 007662 012737 022040 003524 4$:      MOV      #ECCW!MEWD!DMD,E.MR1 ;LOAD EXPECTED MAINT REG. 1
2845 007670 005037 003622          CLR     BITCNT         ;INITIALIZE BIT COUNT
2846 007674 012700 000310          MOV      #200,RO       ;ISSUE 200 MAINT BITS
2847 007700 012762 000440 000026 5$:      MOV      #DMD!MCLK,RKMR1(R2)
2848 007706 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2849 007714 012762 000440 000026      MOV      #DMD!MCLK,RKMR1(R2)
2850 007722 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2851 007730 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MAINT REG. 1
2852 007736 023737 003524 003464      CMP      E.MR1,T.MR1    ;CHECK MAINTENANCE REG. 1 CORRECT
2853 007744 001403          BEQ     6$              ;YES, CONTINUE
2854 007746 104025          ERROR  25              ;MAINT REG 1 INCORRECT
2855 007750 000137 010550          JMP     60$            ;CLEAR RK611
2856
2857 007754 005300          DEC     RO              ;CHECK IF READY FOR SECTOR PULSE
2858 007756 001350          BNE     5$              ;NO, GET NEXT BIT
2859 007760 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2860 007766 016237 000004 003444      MOV      RKBA(R2),T.BA   ;STORE BUS ADDRESS REG
2861 007774 016237 000002 003442      MOV      RKWC(R2),T.WC   ;STORE WORD COUNT REG
2862 010002 023737 003500 003440      CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
2863 010010 001403          BEQ     10$             ;YES, CONTINUE
2864 010012 104026          ERROR  26              ;CS1 INCORRECT
2865 010014 000137 010550          JMP     60$            ;CLEAR RK611
2866
2867 010020 023737 003504 003444 10$:     CMP      E.BA,T.BA      ;CHECK BUS ADDRESS CORRECT
2868 010026 001403          BEQ     11$             ;YES, CONTINUE
2869 010030 104027          ERROR  27              ;BUS ADDRESS INCORRECT
2870 010032 000137 010550          JMP     60$            ;CLEAR RK611
2871
```

```

2872 010036 023737 003502 003442 11$: CMP E.WC,T.WC :CHECK WORD COUNT REG. CORRECT
2873 010044 001403 BEQ 12$ :YES, CONTINUE
2874 010046 104030 ERROR 30 :WORD COUNT INCORRECT
2875 010050 000137 010550 JMP 60$ :CLEAR RK611
2876
2877 010054 012700 000004 12$: MOV #4,RO :SIMULATE SECTOR PULSE
2878 010060 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2)
2879 010066 012762 000540 000026 13$: MOV #DMD!MSP!MCLK,RKMR1(R2)
2880 010074 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2)
2881 010102 005300 DEC RO
2882 010104 001370 BNE 13$
2883 010106 012762 000040 000026 MOV #DMD,RKMR1(R2)
2884 010114 016237 000000 003440 MOV RKCS1(R2),T.CS1 :STORE COMMAND AND STATUS REG. 1
2885 010122 016237 000004 003444 MOV RKBA(R2),T.BA :STORE BUS AND REG.
2886 010130 016237 000002 003442 MOV RKWC(R2),T.WC :STORE WORD COUNT REG.
2887 010136 023737 003500 003440 CMP E.CS1,T.CS1 :CHECK COMMAND AND STATUS REG. 1 INCORRECT
2888 010144 001402 BEQ 15$ :YES, CONTINUE
2889 010146 104026 ERROR 26 :CS1 INCORRECT
2890 010150 000577 BR 60$ :CLEAR RK611
2891
2892 010152 023737 003504 003444 15$: CMP E.BA,T.BA :CHECK BUS ADDRESS CORRECT
2893 010160 001402 BEQ 16$ :YES, CONTINUE
2894 010162 104027 ERROR 27 :BUS ADDRESS INCORRECT
2895 010164 000571 BR 60$ :CLEAR RK611
2896
2897 010166 023737 003502 003442 16$: CMP E.WC,T.WC :CHECK WORD COUNT CORRECT
2898 010174 001402 BEQ 20$ :YES, CONTINUE
2899 010176 104030 ERROR 30 :WORD COUNT INCORRECT
2900 010200 000563 BR 60$ :CLEAR RK611
2901
2902 010202 005037 003622 20$: CLR BITCNT :INITIALIZE BIT COUNT
2903 010206 012700 000310 MOV #200,RO :ISSUE 200 MAINT BITS
2904 010212 012762 000440 000026 21$: MOV #DMD!MCLK,RKMR1(R2)
2905 010220 012762 000040 000026 MOV #DMD,RKMR1(R2)
2906 010226 012762 000440 000026 MOV #DMD!MCLK,RKMR1(R2)
2907 010234 012762 000040 000026 MOV #DMD,RKMR1(R2)
2908 010242 016237 000026 003464 MOV RKMR1(R2),T.MR1 :STORE MAINT REG. 1
2909 010250 023737 003524 003464 CMP E.MR1,T.MR1 :CHECK MAINT REG. 1 CORRECT
2910 010256 001402 BEQ 22$ :YES, CONTINUE
2911 010260 104025 ERROR 25 :MAINT REG. 1 INCORRECT
2912 010262 000532 BR 60$ :CLEAR RK611
2913
2914 010264 005300 22$: DEC RO :CHECK IF READY FOR INDEX PULSE
2915 010266 001351 BNE 21$ :NO, GET NEXT BIT
2916 010270 016237 000000 003440 MOV RKCS1(R2),T.CS1 :STORE COMMAND AND STATUS REG. 1
2917 010276 016237 000004 003444 MOV RKBA(R2),T.BA :STORE BUS ADDRESS
2918 010304 016237 000002 003442 MOV RKWC(R2),T.WC :STORE WORD COUNT
2919 010312 023737 003500 003440 CMP E.CS1,T.CS1 :CHECK CS1 CORRECT
2920 010320 001402 BEQ 23$ :YES, CONTINUE
2921 010322 104026 ERROR 26 :CS1 INCORRECT
2922 010324 000511 BR 60$ :CLEAR RK611
2923
2924 010326 023737 003504 003444 23$: CMP E.BA,T.BA :CHECK BUS ADDRESS CORRECT
2925 010334 001402 BEQ 24$ :YES, CONTINUE
2926 010336 104027 ERROR 27 :BUS ADDRESS INCORRECT
2927 010340 000503 BR 60$ :CLEAR RK611

```

CZ  
CZ



```
2928  
2929 010342 023737 003502 003442 24$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT  
2930 010350 001402 BEQ 25$ ;YES, CONTINUE  
2931 010352 104030 ERROR 30 ;WORD COUNT INCORRECT  
2932 010354 000475 BR 60$ ;CLEAR RK611  
2933  
2934 010356 012762 000240 000026 25$: MOV #DMD!MIND,RKMR1(R2) ;SIMULATE PULSE  
2935 010364 012700 000004 MOV #4,R0  
2936 010370 012762 000640 000026 26$: MOV #DMD!MIND!MCLK,RKMR1(R2)  
2937 010376 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)  
2938 010404 005300 DEC R0  
2939 010406 001370 BNE 26$  
2940 010410 012762 000040 000026 MOV #DMD,RKMR1(R2)  
2941 010416 012700 000002 MOV #2,R0 ;SIMULATE TWO CLOCK PULSES FOR WRITE  
2942 ; GATE TO COME UP  
2943 010422 012762 000440 000026 27$: MOV #DMD!MCLK,RKMR1(R2)  
2944 010430 012762 000040 000026 MOV #DMD,RKMR1(R2)  
2945 010436 005300 DEC R0  
2946 010440 001370 BNE 27$  
2947 010442 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG.  
2948 010450 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1  
2949 010456 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS  
2950 010464 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT  
2951 010472 012737 062040 003524 MOV #WRTGAT!MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MAINT REG. 1  
2952 010500 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT  
2953 010506 001401 BEQ 28$ ;YES, CONTINUE  
2954 010510 104031 ERROR 31 ;CS1 INCORRECT  
2955 010512 023737 003504 003444 28$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT  
2956 010520 001401 BEQ 29$ ;YES, CONTINUE  
2957 010522 104032 ERROR 32 ;BUS ADDRESS INCORRECT  
2958 010524 023737 003502 003442 29$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT  
2959 010532 001401 BEQ 30$ ;YES, CONTINUE  
2960 010534 104033 ERROR 33 ;WORD COUNT INCORRECT  
2961 010536 023737 003524 003464 30$: CMP E.MR1,T.MR1 ;CHECK MAINT REG 1 CORRECT  
2962 010544 001401 BEQ 60$ ;YES, CLEAR RK611  
2963 010546 104034 ERROR 34 ;MAINT REG. 1 INCORRECT  
2964 010550 012762 100000 000000 60$: MOV #CCLR,RKCS1(R2) ;CLEAR RK611
```

.SBTTL \*\*NPR READING OF MEMORY

```
*****  
:TEST 11 NPR OUTPUT DATA TRANSFER  
*****  
: *  
: * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
: * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06  
: * IN 26 SECTOR FORMAT, CYLINDER 777, HEAD 7, DRIVE 7.  
: * SPECIFY A ONE WORD DATA TRANSFER. CLOCK BOTH SEEK  
: * AND DRIVE CLEAR MESSAGES. SIMULATE INDEX PULSE.  
: * CLOCK IN FIRST WORD OF NPR TRANSFER. CHECK INPUT READY,  
: * OUTPUT READY, BUS ADDRESS, WORD COUNT, AND CONTENTS OF  
: * THE SILO. REPEAT FOR 8 DIFFERENT DATA PATTERNS.  
*****
```

```
2980  
2981 010556 000004 TST11: SCOPE  
2982 010560 012737 000144 001200 MOV #100,$TIMES ;DO 100. ITERATIONS  
2983 010566 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
```

```

2984 010572 012703 064310      MOV      #PATTERN,R3      ;LOAD PATTERN ADDRESS
2985 010576 012704 000010      MOV      #8.,R4          ;LOAD PATTERN COUNT
2986 010602 012737 064632 003504  MOV      #WRBUFF+2,E.BA  ;LOAD EXPECTED BUS ADDRESS
2987 010610 012737 000027 003500  MOV      #WRHEAD,E.CS1   ;LOAD EXPECTED CS1
2988 010616 005037 003502      CLR      E.WC           ;LOAD EXPECTED WORD COUNT
2989 010622 012737 010630 001110  MOV      #1$,SLPERR      ;LOAD LOOP ON ERROR LOCATION FOR
2990                                     ; SUBTEST LOOP
2991
2992 010630                                     1$:
2993 010630 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2994 010636 011337 064630      MOV      (R3),WRBUFF     ;LOAD BUFFER FOR WRITE HEADER
2995 010642 012762 000040 000026  MOV      #DMD,RKMR1(R2)  ;PUT RK611 IN MAINT MODE
2996 010650 012762 000777 000020  MOV      #777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS
2997 010656 012762 003400 000006  MOV      #3400,RKDA(R2)  ;LOAD DISK ADDRESS
2998 010664 012762 064630 000004  MOV      #WRBUFF,RKBA(R2) ;LOAD BUS ADDRESS
2999 010672 012762 000007 000010  MOV      #7,RKCS2(R2)    ;LOAD OTHER NUMBER
3000 010700 012762 177777 000002  MOV      #-1,RKWC(R2)    ;LOAD WORD COUNT
3001 010706 012762 000027 000000  MOV      #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
3002 010714 012700 000312      MOV      #50.+4+2,R0     ;ISSUE CLOCKS UNTIL READY FOR
3003                                     ; INDEX PULSE
3004 010720 012762 000440 000026  2$:  MOV      #DMD!MCLK,RKMR1(R2)
3005 010726 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3006 010734 005300      DEC      R0
3007 010736 001370      BNE     2$
3008 010740 012700 000004      MOV      #4,R0          ;SIMULATE INDEX PULSE
3009 010744 012762 000240 000026  MOV      #DMD!MIND,RKMR1(R2)
3010 010752 012762 000640 000026  3$:  MOV      #DMD!MIND!MCLK,RKMR1(R2)
3011 010760 012762 000240 000026  MOV      #DMD!MIND,RKMR1(R2)
3012 010766 005300      DEC      R0
3013 010770 001370      BNE     3$
3014 010772 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3015 011000 012700 000045      MOV      #37.,R0        ;SIMULATE 1 NPR TRANSFER
3016 011004 012762 000440 000026  4$:  MOV      #DMD!MCLK,RKMR1(R2)
3017 011012 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3018 011020 005300      DEC      R0
3019 011022 001370      BNE     4$
3020 011024 016237 000000 003440  MOV      RKCS1(R2),T.CS1  ;STORE COMMAND AND STATUS REG. 1
3021 011032 016237 000004 003444  MOV      RKBA(R2),T.BA   ;STORE BUS ADDRESS REG
3022 011040 016237 000002 003442  MOV      RKWC(R2),T.WC   ;STORE WORD COUNT
3023 011046 012700 000024      MOV      #20.,R0        ;WAIT FOR OUTPUT READY
3024 011052 005300      DEC      R0
3025 011054 001376      BNE     5$
3026 011056 016237 000010 003450  MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3027 011064 012737 000307 003510  MOV      #OR!R!7,E.CS2  ;LOAD EXPECTED CS2
3028 011072 023737 003500 003440  CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
3029 011100 001401      BEQ     6$              ;YES, CHECK CS2
3030 011102 104035      ERROR  35              ;CS1 INCORRECT
3031 011104 023737 003510 003450  6$:  CMP      E.CS2,T.CS2    ;CHECK CS2 CORRECT
3032 011112 001401      BEQ     7$              ;YES, CONTINUE
3033 011114 104036      ERROR  36              ;CS2 INCORRECT
3034 011116 023737 003504 003444  7$:  CMP      E.BA,T.BA     ;CHECK IF BUS ADDRESS INCREMENT OCCURRED
3035 011124 001401      BEQ     8$              ;YES, CONTINUE
3036 011126 104037      ERROR  37              ;BUS ADDRESS INCORRECT
3037 011130 023737 003502 003442  8$:  CMP      E.WC,T.WC     ;CHECK WORD COUNT REG CORRECT
3038 011136 001401      BEQ     9$              ;YES, CONTINUE
3039 011140 104040      ERROR  40              ;WORD COUNT INCORRECT
  
```

```

3040 011142 016237 000024 003462 9$: MOV RKDB(R2),T.DB ;READ DATA BUFFER
3041 011150 011337 003522      MOV (R3),E.DB ;LOAD EXPECTED DATA BUFFER
3042 011154 023737 003522 003462  CMP E.DB,T.DB ;CHECK IF DATA CORRECT
3043 011162 001401      BEQ 15$ ;YES,CONTINUE
3044 011164 104041      ERROR 41 ;DATA BUFFER INCORRECT
3045 011166 016237 000000 003440 15$: MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3046 011174 016237 000010 003450  MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3047 011202 012737 000107 003510  MOV #IR!7,E.CS2 ;LOAD EXPECTED CS2
3048 011210 023737 003500 003440  CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
3049 011216 001401      BEQ 17$ ;YES, CONTINUE
3050 011220 104042      ERROR 42 ;CS1 INCORRECT
3051 011222 023737 003510 003450 17$: CMP E.CS2,T.CS2 ;CHECK COMMAND AND STATUS REG. 2 CORRECT
3052 011230 001401      BEQ 20$ ;YES, CONTINUE
3053 011232 104043      ERROR 43 ;CS2 INCORRECT
3054 011234 104415      20$: SCOP1 ;CHECK IF LOOP ON ERROR
3055 011236 005723      TST (R3)+ ;GENERATE ADDRESS OF NEXT CONFIG
3056 011240 005304      DEC R4 ;CHECK IF ALL 8 CONFIGS TRIED
3057 011242 001000      BNE TST12 ;NO, TRY NEXT DATA PATTERN
  
```

\*\*\*\*\*  
 \*TEST 12 PARTIAL SILO FILLING  
 \*\*\*\*\*

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0. SPECIFY A ONE WORD DATA TRANSFER. CLOCK IN ALL SPECIFIED WORDS INTO THE SILO. CHECK WORD COUNT, BUS ADDRESS, INPUT READY, AND OUTPUT READY. MAKE SURE NO MORE THAN SPECIFIED DATA LENGTH IS CLOCKED INTO THE SILO. CHECK THE SILO FOR CORRECT DATA. REPEAT FOR WORD COUNTS 2-65.

```

3072
3073 011244 000004      TS112: SCOPE
3074 011246 012737 000144 001200  MOV #100, $TIMES ;:DO 100. ITERATIONS
3075 011254 013702 001270      MOV $BASE,R2 ;:LOAD RK611 BASE
3076 011260 012737 000001 001160  MOV #1,$TMP0 ;:LOAD NUMBER OF WORDS FOR DATA TRANSFER
3077 011266 012704 000065      MOV #65,R4 ;:LOAD ITERATION COUNT
3078 011272 012737 000027 003500  MOV #WRHEAD,E.CS1 ;:LOAD EXPECTED CS1
3079 011300 012737 011306 001110  MOV #1$, $LPERR ;:LOAD LOOP ON ERROR LOCATION FOR
3080                                     ; SUBTEST LOOP
3081
3082 011306      1$: MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
3083 011306 012762 100000 000000  MOV #DMD,RKMR1(R2) ;:PUT RK611 IN MAINT MODE
3084 011314 012762 000040 000026  MOV #NPRBUF,RKBA(R2) ;:LOAD BUS ADDRESS
3085 011322 012762 064406 000004  MOV #NPRBUF,E.BA
3086 011330 012737 064406 003504  MOV $TMP0,E.WC ;:LOAD WORD COUNT
3087 011336 013737 001160 003502  NEG E.WC
3088 011344 005437 003502      MOV E.WC,RKWC(R2)
3089 011350 013762 003502 000002  MOV #WRHEAD,RKCS1(R2) ;:ISSUE WRITE HEADER
3090 011356 012762 000027 000000  MOV #50.*4+2,R0 ;:ISSUE CLOCKS UNTIL READY FOR
3091 011364 012700 000312      ; INDEX PULSE
3092
3093 011370 012762 000440 000026 2$: MOV #DMD!MCLK,RKMR1(R2)
3094 011376 012762 000040 000026  MOV #DMD,RKMR1(R2)
3095 011404 005300      DEC R0
  
```

```

3096 011406 001370 BNE 2$
3097 011410 012700 000004 MOV #4,RO ;SIMULATE INDEX PULSE
3098 011414 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
3099 011422 012762 000640 000026 3$: MOV #DMD!MIND!MCLK,RKMR1(R2)
3100 011430 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
3101 011436 005300 DEC RO
3102 011440 001370 BNE 3$
3103 011442 012762 000040 000026 MOV #DMD,RKMR1(R2)
3104 011450 012737 000300 003510 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3105 011456 012700 000045 4$: MOV #37,RO ;SIMULATE 1 NPR TRANSFER
3106 011462 012762 000440 000026 5$: MOV #DMD!MCLK,RKMR1(R2)
3107 011470 012762 000040 000026 MOV #DMD,RKMR1(R2)
3108 011476 005300 DEC RO
3109 011500 001370 BNE 5$
3110 011502 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3111 011510 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
3112 011516 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT
3113 011524 022737 064406 003504 CMP #NPRBUF,E.BA ;CHECK IF FIRST WORD
3114 011532 001004 BNE 7$ ;NO, STORE CS2
3115 011534 012700 000024 MOV #20.,RO ;WAIT FOR OUTPUT READY
3116 011540 005300 6$: DEC RO
3117 011542 001376 BNE 6$
3118 011544 016237 000010 003450 7$: MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3119 011552 062737 000002 003504 ADD #2,E.BA ;INCREMENT WORD COUNT AND BUS ADD
3120 011560 005237 003502 INC E.WC
3121 011564 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
3122 011572 001401 BEQ 8$ ;YES, CHECK CS2
3123 011574 104044 ERROR 44 ;CS1 INCORRECT
3124 011576 023737 003510 003450 8$: CMP E.CS2,T.CS2 ;CHECK COMMAND STATUS REG. 2 CORRECT
3125 011604 001401 BEQ 9$ ;YES, CHECK BUSS ADDRESS REG.
3126 011606 104045 ERROR 45 ;CS2 INCORRECT
3127 011610 023737 003504 003444 9$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
3128 011616 001401 BEQ 10$ ;YES, CHECK WORD COUNT
3129 011620 104046 ERROR 46 ;BUS ADDRESS INCORRECT
3130 011622 023737 003502 003442 10$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
3131 011630 001401 BEQ 11$ ;YES, CHECK IF ALL WORDS TRANSFERRED
3132 011632 104047 ERROR 47 ;WORD COUNT INCORRECT
3133 011634 005737 003502 11$: TST E.WC ;CHECK IF FINISHED
3134 011640 001306 BNE 4$ ;NO, TRANSFER NEXT WORD
3135 011642 012700 000112 MOV #2*37.,RO ;ISSUE ENOUGH CLOCKS FOR
3136 011646 012762 000440 000026 15$: MOV #DMD!MCLK,RKMR1(R2) ; 2 NPR TRANSFERS
3137 011654 012762 000040 000026 MOV #DMD,RKMR1(R2)
3138 011662 005300 DEC RO
3139 011664 001370 BNE 15$
3140 011666 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3141 011674 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3142 011702 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS REG.
3143 011710 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT
3144 011716 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
3145 011724 001401 BEQ 16$ ;YES, CHECK CS2
3146 011726 104050 ERROR 50 ;CS1 INCORRECT
3147 011730 023737 003510 003450 16$: CMP E.CS2,T.CS2 ;CHECK COMMAND STATUS REG. 2 CORRECT
3148 011736 001401 BEQ 17$ ;YES, CHECK BUS ADDRESS
3149 011740 104051 ERROR 51 ;CS2 INCORRECT
3150 011742 023737 003504 003444 17$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
3151 011750 001401 BEQ 18$ ;YES, CHECK WORD COUNT
  
```



```

3208 012230 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
3209 012236 012762 000040 000026 MOV #DMD,RKMR1(R2)
3210 012244 005300 DEC RO
3211 012246 001370 BNE 1$
3212 012250 012700 000004 MOV #4,RO ;SIMULATE INDEX PULSE
3213 012254 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
3214 012262 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
3215 012270 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
3216 012276 005300 DEC RO
3217 012300 001370 BNE 2$
3218 012302 012762 000040 000026 MOV #DMD,RKMR1(R2)
3219 012310 012737 000300 003510 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3220 012316 012700 000045 4$: MOV #37,RO ;SIMULATE 1 NPR TRANSFER
3221 012322 012762 000440 000026 5$: MOV #DMD!MCLK,RKMR1(R2)
3222 012330 012762 000040 000026 MOV #DMD,RKMR1(R2)
3223 012336 005300 DEC RO
3224 012340 001370 BNE 5$
3225 012342 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3226 012350 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
3227 012356 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT
3228 012364 022737 064410 003504 CMP #NPRBUF+2,E.BA ;CHECK IF FIRST WORD
3229 012372 001004 BNE 7$ ;NO, STORE CS2
3230 012374 012700 000024 MOV #20,RO ;WAIT FOR OUTPUT READY
3231 012400 005300 6$: DEC RO
3232 012402 001376 BNE 6$
3233 012404 016237 000010 003450 7$: MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3234 012412 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
3235 012420 001401 BEQ 8$ ;YES, CHECK CS2
3236 012422 104044 ERROR 44 ;CS1 INCORRECT
3237 012424 023737 003510 003450 8$: CMP E.CS2,T.CS2 ;CHECK COMMAND STATUS REG. 2 CORRECT
3238 012432 001401 BEQ 9$ ;YES, CHECK BUS ADDRESS
3239 012434 104045 ERROR 45 ;CS2 INCORRECT
3240 012436 023737 003504 003444 9$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
3241 012444 001401 BEQ 10$ ;YES, CHECK WORD COUNT
3242 012446 104046 ERROR 46 ;BUS ADDRESS INCORRECT
3243 012450 023737 003502 003442 10$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
3244 012456 001401 BEQ 11$ ;YES, CHECK IF ALL WORDS TRANSFERRED
3245 012460 104047 ERROR 47 ;WORD COUNT INCORRECT
3246 012462 062737 000002 003504 11$: ADD #2,E.BA ;INCREMENT WORD COUNT AND BUS ADDRESS
3247 012470 005237 003502 INC E.WC ;
3248 012474 100710 BMI 4$ ;CHECK IF FINISHED (NO, BRANCH)
3249 012476 001004 BNE 12$ ;CHECK IF LAST WORD
3250 012500 012737 000200 003510 MOV #OR,E.CS2 ;LOAD EXPECTED CS2
3251 012506 000703 BR 4$ ;PROCESS LAST WORD
3252
3253 012510 005037 003502 12$: CLR E.WC ;ADJUST EXPECTED WORD COUNT
3254 012514 162737 000002 003504 SUB #2,E.BA ; AND BUS ADDRESS
3255 012522 012700 000112 MOV #2+37,RO ;ISSUE ENOUGH CLOCKS FOR
3256 012526 012762 000440 000026 15$: MOV #DMD!MCLK,RKMR1(R2) ; 2 NPR TRANSFERS
3257 012534 012762 000040 000026 MOV #DMD,RKMR1(R2)
3258 012542 005300 DEC RO
3259 012544 001370 BNE 15$
3260 012546 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3261 012554 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3262 012562 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS REG
3263 012570 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT
  
```

```

3264 012576 023737 003500 003440      CMP      E.CS1,T.CS1      :CHECK COMMAND STATUS REG. 1 CORRECT
3265 012604 001401      BEQ      16$            :YES, CHECK CS2
3266 012606 104050      ERROR   50            :CS1 INCORRECT
3267 012610 023737 003510 003450 16$:      CMP      E.CS2,T.CS2      :CHECK COMMAND STATUS REG. 2 CORRECT
3268 012616 001401      BEQ      17$            :YES, CHECK BUS ADDRESS
3269 012620 104051      ERROR   51            :CS2 INCORRECT
3270 012622 023737 003504 003444 17$:      CMP      E.BA,T.BA      :CHECK BUS ADDRESS CORRECT
3271 012630 001401      BEQ      18$            :YES, CHECK WORD COUNT
3272 012632 104052      ERROR   52            :BUS ADDRESS INCORRECT
3273 012634 023737 003502 003442 18$:      CMP      E.WC,T.WC      :CHECK WORD COUNT CORRECT
3274 012642 001401      BEQ      19$            :YES, CHECK DATA
3275 012644 104053      ERROR   53            :WORD COUNT INCORRECT
3276 012646 012701 000102      MOV      #66.,R1        :LOAD NUMBER OF WORDS LOADED
3277 012652 012703 064406      MOV      #NPRBUF,R3     :LOAD START ADDRESS
3278 012656 005037 003624      CLR      WRDCNT         :INITIALIZE WORD COUNT FOR PRINT OUT
3279 012662 012737 000300 003510      MOV      #IR!OR,E.CS2   :LOAD EXPECTED CS2
3280 012670 016237 000024 003462 25$:      MOV      RKDB(R2),T.DB  :READ DATA BUFFER
3281 012676 012337 003522      MOV      (R3)+,E.DB     :LOAD EXPECTED DATA BUFFER
3282 012702 023737 003522 003462      CMP      E.DB,T.DB     :CHECK IF DATA CORRECT
3283 012710 001401      BEQ      26$            :YES, CONTINUE
3284 012712 104054      ERROR   54            :DATA BUFFER INCORRECT
3285 012714 016237 000000 003440 26$:      MOV      RKCS1(R2),T.CS1 :STORE COMMAND AND STATUS REG 1
3286 012722 005737 003624      TST      WRDCNT         :CHECK IF FIRST WORD
3287 012726 001015      BNE      28$            :NO, GET CS2
3288 012730 012700 000024      MOV      #20.,R0        :WAIT FOR INPUT READY TO SET
3289 012734 005300 27$:      DEC      R0
3290 012736 001376      BNE      27$
3291 012740 016237 000010 003450      MOV      RKCS2(R2),T.CS2 :STORE COMMAND AND STATUS REG. 2
3292 012746 022701 000001      CMP      #1,R1          :CHECK IF LAST WORD IN SILO
3293 012752 001003      BNE      28$            :NO, CONTINUE
3294 012754 012737 000100 003510      MOV      #IR,E.CS2     :LOAD EXPECTED CS2
3295 012762 023737 003500 003440 28$:      CMP      E.CS1,T.CS1   :CHECK COMMAND AND STATUS REG. 1 CORRECT
3296 012770 001401      BEQ      29$            :YES, CHECK CS2
3297 012772 104055      ERROR   55            :CS1 INCORRECT
3298 012774 023737 003510 003450 29$:      CMP      E.CS2,T.CS2   :CHECK COMMAND AND STATUS REG. 2 CORRECT
3299 013002 001401      BEQ      30$            :YES, GET NEXT WORD
3300 013004 104056      ERROR   56            :CS2 INCORRECT
3301 013006 005237 003624 30$:      INC      WRDCNT        :INCREMENT WORD COUNT
3302 013012 005301      DEC      R1             :DECREMENT WORDS READ
3303 013014 001325      BNE      25$           :CHECK IF ALL WORDS READ
    
```

```

*****
:TEST 14      SILO CAPICITY WITH NPR TRANSFERS
:
:  CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
:  IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
:  IN 26 SECTOR FORMAT, CYLINDER 0.  HEAD 0, DRIVE 0.
:  SPECIFY A 68 WORD DATA TRANSFER.  CLOCK IN 66
:  WORDS INTO THE SILO.  MAKE SURE THAT SILO WILL STOP
:  FILLING AT 66 WORDS.  TAKE ONE WORD FROM SILO AND
:  CHECK IT.  CLOCK IN NEXT WORD.  MAKE SURE NO MORE
:  THAN ONE WORD IS CLOCKED IN THE SILO.  TAKE ONE
:  WORD FROM SILO AND CHECK IT.  CLOCK IN NEXT WORD.
:  CLOCK IN NEXT WORD.  MAKE SURE NO MORE THAN ONE WORD IS
:  CLOCKED IN THE SILO.  TAKE ONE WORD FROM SILO AND
:  CHECK IT.  ATTEMPT TO CLOCK IN NEXT WORD AND
    
```

3304  
3305  
3306  
3307  
3308  
3309  
3310  
3311  
3312  
3313  
3314  
3315  
3316  
3317  
3318  
3319

```
3320      : * MAKE SURE NO WORDS ARE CLOCK INTO SILO. UNLOAD THE
3321      : * SILO AND MAKE SURE ALL THE WORDS ARE CORRECT.
3322      : *
3323      : *
3324      : *****
3324 013016 000004 TST14: SCOPE
3325 013020 012737 000144 001200 MOV #100.,$TIMES ;:DO 100. ITERATIONS
3326 013026 013702 001270 MOV $BASE,R2 ;:LOAD RK611 BASE
3327 013032 012737 000027 003500 MOV #WRHEAD,E.CS1 ;:LOAD EXPECTED CS1
3328 013040 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
3329 013046 012762 000040 000026 MOV #DMD,RKMR1(R2) ;:PUT RK611 IN MAINT MODE
3330 013054 012737 064410 003504 MOV #NPRBUF+2,E.BA ;:LOAD BUS ADDRESS
3331 013062 012762 064406 000004 MOV #NPRBUF,RKBA(R2)
3332 013070 012737 177675 003502 MOV #-67.,E.WC ;:LOAD WORD COUNT
3333 013076 012762 177674 000002 MOV #-68.,RKWC(R2)
3334 013104 012762 000027 000000 MOV #WRHEAD,RKCS1(R2) ;:ISSUE WRITE HEADER
3335 013112 012700 000312 MOV #50.*4+2,R0 ;:ISSUE CLOCKS UNTIL READY FOR
3336      : : INDEX PULSE
3337 013116 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
3338 013124 012762 000040 000026 MOV #DMD,RKMR1(R2)
3339 013132 005300 DEC R0
3340 013134 001370 BNE 1$
3341 013136 012700 000004 MOV #4,R0 ;:SIMULATE INDEX PULSE
3342 013142 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
3343 013150 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
3344 013156 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
3345 013164 005300 DEC R0
3346 013166 001370 BNE 2$
3347 013170 012737 000040 000026 MOV #DMD,RKMR1
3348 013176 012737 000300 003510 MOV #IR!OR,E.CS2 ;:LOAD EXPECTED CS2
3349 013204 012700 000045 000026 4$: MOV #37.,R0 ;:SIMULATE 1 NPR TRANSFER
3350 013210 012762 000440 000026 5$: MOV #DMD!MCLK,RKMR1(R2)
3351 013216 012762 000040 000026 MOV #DMD,RKMR1(R2)
3352 013224 005300 DEC R0
3353 013226 001370 BNE 5$
3354 013230 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;:STORE COMMAND AND REG. 1
3355 013236 016237 000004 003444 MOV RKBA(R2),T.BA ;:STORE BUS ADDRESS
3356 013244 016237 000002 003442 MOV RKWC(R2),T.WC ;:STORE WORD COUNT
3357 013252 022737 064410 003504 CMP #NPRBUF+2,E.BA ;:CHECK IF FIRST WORD
3358 013260 001004 BNE 7$ ;:NO, STORE CS2
3359 013262 012700 000024 MOV #20.,R0 ;:WAIT FOR OUTPUT READY
3360 013266 005300 6$: DEC R0
3361 013270 001376 BNE 6$
3362 013272 016237 000010 003450 7$: MOV RKCS2(R2),T.CS2 ;:STORE COMMAND AND STATUS REG. 2
3363 013300 023737 003500 003440 CMP E.CS1,T.CS1 ;:CHECK COMMAND STATUS REG. 1 CORRECT
3364 013306 001401 BEQ 8$ ;:YES, CHECK CS2
3365 013310 104044 ERROR 44 ;:CS1 INCORRECT
3366 013312 023737 003510 003450 8$: CMP E.CS2,T.CS2 ;:CHECK COMMAND STATUS REG. 2 CORRECT
3367 013320 001401 BEQ 9$ ;:YES, CHECK BUS ADDRESS
3368 013322 104045 ERROR 45 ;:CS2 INCORRECT
3369 013324 023737 003504 003444 9$: CMP E.BA,T.BA ;:CHECK BUS ADDRESS CORRECT
3370 013332 001401 BEQ 10$ ;:YES, CHECK WORD COUNT
3371 013334 104046 ERROR 46 ;:BUS ADDRESS INCORRECT
3372 013336 023737 003502 003442 10$: CMP E.WC,T.WC ;:CHECK WORD COUNT CORRECT
3373 013344 001401 BEQ 11$ ;:YES, CHECK IF 1ST 65 WORDS TRANSFERRED
3374 013346 104047 ERROR 47 ;:WORD COUNT INCORRECT
3375 013350 062737 000002 003504 11$: ADD #2,E.BA ;:INCREMENT BUS ADD AND WORD COUNT
```



3376	013356	005237	003502			INC	E.WC	
3377	013362	022737	177776	003502		CMP	#-2,E.WC	:CHECK IF 65 WORDS IN SILO
3378	013370	101305				BHI	48	:NO, GET NEXT WORD
3379	013372	001004				BNE	128	:CHECK IF ALL 65 WORDS IN SILO
3380	013374	012737	000200	003510		MOV	#OR,E.CS2	:LOAD EXPECTED CS2
3381	013402	000700				BR	48	:PROCESS 66TH WORD
3382								
3383	013404	005337	003502		128:	DEC	E.WC	:ADJUST WORD COUNT AND
3384	013410	162737	000002	003504		SUB	#2,E.BA	: BUS ADDRESS
3385	013416	012701	000003			MOV	#3,R1	
3386	013422	005037	003624			CLR	WRDCNT	
3387	013426	012703	064406			MOV	#NPRBUF,R3	:LOAD START ADDRESS
3388	013432	012700	000112		138:	MOV	#2+37.,R0	:ISSUE ENOUGH CLOCKS FOR
3389	013436	012762	000440	000026	158:	MOV	#DMD!MCLK,RKMR1(R2)	: 2 NPR TRANSFERS
3390	013444	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3391	013452	005300				DEC	R0	
3392	013454	001370				BNE	158	
3393	013456	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3394	013464	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3395	013472	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG
3396	013500	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT
3397	013506	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1
3398	013514	001401				BEQ	168	:YES, CHECK CS2
3399	013516	104050				ERROR	50	:CS1 INCORRECT
3400	013520	023737	003510	003450	168:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3401	013526	001401				BEQ	178	:YES,CHECK BUS ADDRESS
3402	013530	104051				ERROR	51	:CS2 INCORRECT
3403	013532	023737	003504	003444	178:	CMP	E.BA,T.BA	:CHECK BUS ADD CORRECT
3404	013540	001401				BEQ	188	:YES, CHECK WORD COUNT
3405	013542	104052				ERROR	52	:BUS ADDRESS INCORRECT
3406	013544	023737	003502	003442	188:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3407	013552	001401				BEQ	198	:YES, READ 1 WORD FROM SILO
3408	013554	104053				ERROR	53	:WORD COUNT INCORRECT
3409	013556	012737	000300	003510	198:	MOV	#IR!OR,E.CS2	:LOAD EXPECT CS2
3410	013564	016237	000024	003462		MOV	RKDB(R2),T.DB	:STORE DATA BUFFER
3411	013572	012337	003522			MOV	(R3)+,E.DB	:LOAD EXPECTED DATA BUFFER
3412	013576	023737	003522	003462		CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3413	013604	001401				BEQ	258	:YES, CONTINUE
3414	013606	104054				ERROR	54	:DATA BUFFER INCORRECT
3415	013610	016237	000000	003440	258:	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3416	013616	012700	000024			MOV	#20.,R0	:WAIT FOR OUTPUT READY
3417	013622	005300			268:	DEC	R0	
3418	013624	001376				BNE	268	
3419	013626	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3420	013634	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK IF COMMAND STATUS REG. 1 CORRECT
3421	013642	001401				BEQ	278	:YES, CHECK CS2
3422	013644	104055				ERROR	55	:CS1 INCORRECT
3423	013646	023737	003510	003450	278:	CMP	E.CS2,T.CS2	:CHECK IF COMMAND STATUS REG. 2 CORRECT
3424	013654	001401				BEQ	308	:YES, DO NPR TRANSFER
3425	013656	104056				ERROR	56	:CS2 INCORRECT
3426	013660	012700	000045		308:	MOV	#37.,R0	:CLOCK IN ONE WORD (NPR TRANSFER)
3427	013664	012762	000440	000026	318:	MOV	#DMD!MCLK,RKMR1(R2)	
3428	013672	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3429	013700	005300				DEC	R0	
3430	013702	001370				BNE	318	
3431	013704	022701	000001			CMP	#1,R1	:CHECK IF 68TH WORD READ

3432 013710 001410 BEQ 328 :YES, NO NPR WILL TAKE PLACE  
3433 013712 062737 000002 003504 ADD #2,E.BA :INCREMENT BUS ADD AND WORD COUNT  
3434 013720 005237 003502 INC E.WC  
3435 013724 012737 000200 003510 MOV #OR,E.CS2 :LOAD EXPECTED CS2  
3436 013732 016237 000000 003440 328: MOV RKCS1(R2),T.CS1 :STORE COMMAND AND STATUS REG. 1  
3437 013740 016237 000010 003450 MOV RKCS2(R2),T.CS2 :STORE COMMAND AND STATUS REG. 2  
3438 013746 016237 000004 003444 MOV RKBA(R2),T.BA :STORE BUS ADDRESS  
3439 013754 016237 000002 003442 MOV RKWC(R2),T.WC :STORE WORD COUNT  
3440 013762 023737 003500 003440 CMP E.CS1,T.CS1 :CHECK COMMAND STATUS REG. 1 CORRECT  
3441 013770 001401 BEQ 338 :YES, CHECK CS2  
3442 013772 104044 ERROR 44 :CS1 INCORRECT  
3443 013774 023737 003510 003450 338: CMP E.CS2,T.CS2 :CHECK COMMAND STATUS REG. 2 CORRECT  
3444 014002 001401 BEQ 348 :YES, CHECK BUS ADD  
3445 014004 104045 ERROR 45 :CS2 INCORRECT  
3446 014006 023737 003504 003444 348: CMP E.BA,T.BA :CHECK BUS ADD CORRECT  
3447 014014 001401 BEQ 358 :YES, CHECK WORD COUNT  
3448 014016 104046 ERROR 46 :BUS ADD INCORRECT  
3449 014020 023737 003502 003442 358: CMP E.WC,T.WC :CHECK WORD COUNT CORRECT  
3450 014026 001401 BEQ 368 :YES, CONTINUE  
3451 014030 104047 ERROR 47 :WORD COUNT INCORRECT  
3452 014032 005237 003624 368: INC WRDCNT  
3453 014036 005301 DEC R1 :CHECK IF READ TO UNLOAD SILO  
3454 014040 001402 BEQ 398 :YES, READ DATA; BUFFER  
3455 014042 000137 013432 JMP 138 :NO, INPUT NEXT WORD  
3456  
3457 014046 162737 000002 003504 398: SUB #2,E.BA :ADJUST WORD COUNT AND BUS ADDRESS  
3458 014054 005037 003502 CLR E.WC  
3459 014060 012737 000300 003510 MOV #1R!OR,E.CS2 :LOAD EXPECTED CS2  
3460 014066 012701 000101 MOV #65.,R1 :LOAD NUMBER OF WORDS LEFT  
3461 014072 016237 000024 003462 408: MOV RKDB(R2),T.DB :READ DATA BUFFER  
3462 014100 012337 003522 MOV (R3)+,E.DB :LOAD EXPECTED DATA BUFFER  
3463 014104 023737 003522 003462 CMP E.DB,T.DB :CHECK IF DATA CORRECT  
3464 014112 001401 BEQ 418 :YES, CHECK CS1  
3465 014114 104054 ERROR 54 :DATA BUFFER INCORRECT  
3466 014116 016237 000000 003440 418: MOV RKCS1(R2),T.CS1 :STORE COMMAND AND STATUS REG. 1  
3467 014124 022737 000002 003624 CMP #2,WRDCNT :CHECK IF FIRST WORDS  
3468 014132 001004 BNE 438 :NO, DO NOT WAIT FOR INPUT READY  
3469 014134 012700 000024 MOV #20.,R0 :WAIT FOR INPUT READY  
3470 014140 005300 428: DEC R0  
3471 014142 001376 BNE 428  
3472 014144 016237 000010 003450 438: MOV RKCS2(R2),T.CS2 :STORE COMMAND AND STATUS REG. 2  
3473 014152 022701 000001 CMP #1,R1 :CHECK IF LAST WORD  
3474 014156 001003 BNE 448 :NO, CONTINUE  
3475 014160 012737 000100 003510 MOV #1R,E.CS2 :LOAD EXPECTED CS2  
3476 014166 023737 003500 003440 448: CMP E.CS1,T.CS1 :CHECK CS1 CORRECT  
3477 014174 001401 BEQ 458 :YES,CHECK CS2  
3478 014176 104055 ERROR 55 :CS1 INCORRECT  
3479 014200 023737 003510 003450 458: CMP E.CS2,T.CS2 :CHECK COMMAND AND STATUS REG. 2 CORRECT  
3480 014206 001401 BEQ 468 :YES, READ NEXT WORD ON SILO  
3481 014210 104056 ERROR 56 :CS2 INCORRECT  
3482 014212 005237 003624 468: INC WRDCNT :INCREMENT WORD COUNT  
3483 014216 005301 DEC R1 :CHECK IF ALL WORDS READ  
3484 014220 001324 BNE 408 :NO, READ NEXT WORD

.....  
: TEST 15 BUS ADDRESS INHIBIT

3485  
3486  
3487

3488  
3489  
3490  
3491  
3492  
3493  
3494  
3495  
3496  
3497  
3498  
3499  
3500  
3501  
3502  
3503  
3504  
3505  
3506  
3507  
3508  
3509  
3510  
3511  
3512  
3513  
3514  
3515  
3516  
3517  
3518  
3519  
3520  
3521  
3522  
3523  
3524  
3525  
3526  
3527  
3528  
3529  
3530  
3531  
3532  
3533  
3534  
3535  
3536  
3537  
3538  
3539  
3540  
3541  
3542  
3543

014222	000004		
014224	012737	000144	001200
014232	013702	001270	
014236	012737	000027	003500
014244	012762	100000	000000
014252	012762	000040	000026
014260	012737	064406	003504
014266	012762	064406	000004
014274	012737	177677	003502
014302	012762	177676	000002
014310	012762	000020	000010
014316	012762	000027	000000
014324	012700	000312	
014330	012762	000440	000026
014336	012762	000040	000026
014344	005300		
014346	001370		
014350	012700	000004	
014354	012762	000240	000026
014362	012762	000640	000026
014370	012762	000240	000026
014376	005300		
014400	001370		
014402	012762	000040	000026
014410	012737	000320	003510
014416	012700	000045	
014422	012762	000440	000026
014430	012762	000040	000026
014436	005300		
014440	001370		
014442	016237	000000	003440
014450	016237	000004	003444
014456	016237	000002	003442
014464	022737	000101	003502
014472	001004		
014474	012700	000024	
014500	005300		
014502	001376		
014504	016237	000010	003450
014512	023737	003500	003440
014520	001401		
014522	104057		
014524	023737	003510	003450
014532	001401		
014534	104060		

```

**
** CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
** IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN
** RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
** SPECIFY A 66 WORD DATA TRANSFER WITH BUS ADDRESS
** INHIBIT INCREMENT. CHECK WORD COUNT, BUS ADDRESS,
** INPUT READY, OUTPUT READY, AND MAKE SURE ALL THE
** WORDS IN THE SILO ARE THE CORRECT SAME WORD.
**
*****
TST15: SCOPE
MOV #100.,$TIMES ;;DO 100. ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #WRHEAD,E.CS1 ;LOAD EXPECTED CS1
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE
MOV #NPRBUF,E.BA ;LOAD BUS ADDRESS
MOV #NPRBUF,RKBA(R2)
MOV #-65.,E.WC ;LOAD WORD COUNT
MOV #-66.,RKWC(R2)
MOV #BAI,RKCS2(R2) ;SET BUS ADDRESS INCREMENT INHIBIT
MOV #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
MOV #50.*4+2,R0 ;ISSUE CLOCKS UNTIL READY FOR
; INDEX PULSE
1$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV #4,R0 ;SIMULATE INDEX PULSE
2$: MOV #DMD!MIND,RKMR1(R2)
MOV #DMD!MIND!MCLK,RKMR1(R2)
MOV #DMD!MIND,RKMR1(R2)
DEC R0
BNE 2$
MOV #DMD,RKMR1(R2)
4$: MOV #IR!OR!BAI,E.CS2 ;LOAD EXPECTED CS2
5$: MOV #37.,R0 ;SIMULATE 1 NPR TRANSFER
MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 5$
MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
MOV RKWC(R2),T.WC ;STORE WORD COUNT
CMP #65.,E.WC ;CHECK IF FIRST WORD
BNE 7$ ;NO, STORE CS2
MOV #20.,R0 ;WAIT FOR OUTPUT READY
6$: DEC R0
BNE 6$
7$: MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
CMP E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
BEQ 8$ ;YES, CHECK CS2
ERROR 57 ;CS1 INCORRECT
8$: CMP E.CS2,T.CS2 ;CHECK COMMAND STATUS REG. 2 CORRECT
BEQ 9$ ;YES, CHECK BUS ADDRESS
ERROR 60 ;CS2 INCORRECT
```

CZR6CDO RK611 DSKLS CTRL PRT3 MACY11 30A(1052) 19-JUN-80 13:52 C 6  
CZR6CD.P11 19-JUN-80 13:49 T15 BUS ADDRESS INHIBIT PAGE 68

SEQ 0067

3544	014536	023737	003504	003444	9\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS
3545	014544	001401				BEQ	10\$	:YES, CHECK WORD COUNT
3546	014546	104061				ERROR	61	:BUS ADDRESS INCORRECT
3547	014550	023737	003502	003442	10\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3548	014556	001401				BEQ	11\$	:YES, CHECK IF ALL WORDS TRANSFERRED
3549	014560	104062				ERROR	62	:WORD COUNT INCORRECT
3550	014562	005237	003502		11\$:	INC	E.WC	:INCREMENT WORD COUNT
3551	014566	100713				BMI	4\$	:CHECK IF FINISHED (NO, BRANCH)
3552	014570	001004				BNE	12\$	:CHECK IF LAST WORD
3553	014572	012737	000220	003510		MOV	#OR!BAI,E.CS2	:LOAD EXPECTED COMMAND STATUS REG. 2
3554	014600	000706				BR	4\$	:PROCESS THE LAST WORD
3555								
3556	014602	005037	003502		12\$:	CLR	E.WC	:ADJUST WORD COUNT
3557	014606	012700	000112			MOV	#2*37.,R0	:ISSUE ENOUGH CLOCKS FOR
3558	014612	012762	000440	000026	15\$:	MOV	#DMD!MCLK,RKMR1(R2)	: 2 NPR TRANSFERS
3559	014620	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3560	014626	005300				DEC	R0	
3561	014630	001370				BNE	15\$	
3562	014632	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3563	014640	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3564	014646	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG.
3565	014654	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT REG.
3566	014662	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3567	014670	001401				BEQ	16\$	:YES, CHECK CS2
3568	014672	104063				ERROR	63	:CS1 INCORRECT
3569	014674	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3570	014702	001401				BEQ	17\$	:YES, CHECK BUS ADDRESS
3571	014704	104064				ERROR	64	:CS2 INCORRECT
3572	014706	023737	003504	003444	17\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3573	014714	001401				BEQ	18\$	:YES, CHECK WORD COUNT
3574	014716	104065				ERROR	65	:BUS ADDRESS INCORRECT
3575	014720	023737	003502	003442	18\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3576	014726	001401				BEQ	19\$	:YES, CHECK DATA
3577	014730	104066				ERROR	66	:WORD COUNT INCORRECT
3578	014732	012701	000102		19\$:	MOV	#66.,R1	:LOAD NUMBERS OF WORDS LOADED
3579	014736	013737	064406	003522		MOV	NPRBUF,E.DB	:LOAD EXPECTED DATA BUFFER
3580	014744	005037	003624			CLR	WRDCNT	:INITIALIZE WORD COUNT FOR PRINT OUT
3581	014750	016237	000024	003462	25\$:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER
3582	014756	012737	000320	003510		MOV	#IR!OR!BAI,E.CS2	:LOAD EXPECTED CS2
3583	014764	023737	003522	003462		CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3584	014772	001401				BEQ	26\$	:YES, CONTINUE
3585	014774	104067				ERROR	67	:DATA BUFFER INCORRECT
3586	014776	016237	000000	003440	26\$:	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3587	015004	005737	003624			TST	WRDCNT	:CHECK IF FIRST WORD
3588	015010	001015				BNE	28\$	:NO, GET CS2
3589	015012	012700	000024			MOV	#20.,R0	:WAIT FOR INPUT READY TO SET
3590	015016	005300			27\$:	DEC	R0	
3591	015020	001376				BNE	27\$	
3592	015022	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3593	015030	022701	000001			CMP	#1,R1	:CHECK IF LAST WORD
3594	015034	001003				BNE	28\$	:NO, CONTINUE
3595	015036	012737	000120	003510		MOV	#IR!BAI,E.CS2	:LOAD EXPECTED CS2
3596	015044	023737	003500	003440	28\$:	CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1 CORRECT
3597	015052	001401				BEQ	29\$	:YES, CHECK CS2
3598	015054	104070				ERROR	70	:CS1 INCORRECT
3599	015056	023737	003510	003450	29\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG 2 CORRECT

3600 015064 001401  
3601 015066 104071  
3602 015070 005237 003624  
3603 015074 005301  
3604 015076 001324  
3605  
3606  
3607  
3608  
3609  
3610  
3611  
3612  
3613  
3614  
3615  
3616

30\$: BEQ 30\$ :YES, GET NEXT WORD  
ERROR 71 :CS2 INCORRECT  
INC WRDCNT :INCREMENT WORD COUNT  
DEC R1 :DECREMENT WORDS READ  
BNE 25\$ :CHECK IF ALL WORDS READ

\*\*\*\*\*  
\*TEST 16 NON-EXISTENT MEMORY  
\*  
\* CLEAR RK611 WITH A CONTROLLER CLEAR, PUT CONTROLLER  
\* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06  
\* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
\* SPECIFY A ONE WORD DATA TO A NON-EXISTENT ADDRESS  
\* (760000) AND MAKE SURE THE NON-EXISTENT MEMORY ERROR  
\* OCCURS IN THE RK611 CONTROLLER.  
\*\*\*\*\*

3617 015100 000004  
3618 015102 012737 000144 001200  
3619 015110 013702 001270  
3620 015114 012762 100000 000000  
3621 015122 012762 000040 000026  
3622 015130 012737 160002 003504  
3623 015136 012762 160000 000004  
3624 015144 012737 177677 003502  
3625 015152 012762 177676 000002  
3626 015160 012737 101626 003500  
3627 015166 012762 001427 000000  
3628 015174 012700 000312  
3629  
3630 015200 012762 000440 000026 1\$:  
3631 015206 012762 000040 000026  
3632 015214 005300  
3633 015216 001370  
3634 015220 012700 000004  
3635 015224 012762 000240 000026  
3636 015232 012762 000640 000026 2\$:  
3637 015240 012762 000240 000026  
3638 015246 005300  
3639 015250 001370  
3640 015252 012762 000040 000026  
3641 015260 012700 000045  
3642 015264 012762 000440 000026 5\$:  
3643 015272 012762 000040 000026  
3644 015300 005300  
3645 015302 001370  
3646 015304 016237 000000 003440  
3647 015312 016237 000010 003450  
3648 015320 016237 000004 003444  
3649 015326 016237 000002 003442  
3650 015334 016237 000014 003454  
3651 015342 005037 003514  
3652 015346 012737 004100 003510  
3653 015354 032737 000200 003450  
3654 015362 001403  
3655 015364 052737 000200 003510

TST16: SCOPE  
MOV #100.,STIMES ;;DO 100. ITERATIONS  
MOV \$BASE,R2 ;LOAD RK611 BASE  
MOV #CCLR,RKCS1(R2) ;CLEAR RK611  
MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE  
MOV #160002,E.BA ;LOAD BUS ADDRESS  
MOV #160000,RKBA(R2)  
MOV #-65.,E.WC ;LOAD WORD COUNT  
MOV #-66.,RKWC(R2)  
MOV #CERR!RDY!BA16!BA17!WRHEAD<^C<GO>>,E.CS1  
MOV #BA17!BA16!WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER  
MOV #50.\*4+2,RO ;ISSUE CLOCKS UNTIL READY FOR  
; INDEX PULSE  
1\$: MOV #DMD!MCLK,RKMR1(R2)  
MOV #DMD,RKMR1(R2)  
DEC RO  
BNE 1\$  
MOV #4,RO ;SIMULATE INDEX PULSE  
2\$: MOV #DMD!MIND,RKMR1(R2)  
MOV #DMD!MIND!MCLK,RKMR1(R2)  
MOV #DMD!MIND,RKMR1(R2)  
DEC RO  
BNE 2\$  
MOV #DMD,RKMR1(R2)  
MOV #37,RO ;SIMULATE 1 NPR TRANSFER  
5\$: MOV #DMD!MCLK,RKMR1(R2)  
MOV #DMD,RKMR1(R2)  
DEC RO  
BNE 5\$  
MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1  
MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2  
MOV RKBA(R2),T.BA ;STORE BUS ADDRESS  
MOV RKWC(R2),T.WC ;STORE WORD COUNT  
MOV RKER(R2),T.ER ;STORE ERROR REG.  
CLR E.ER ;LOAD EXPECTED ERROR REG.  
MOV #IR!NEM,E.CS2 ;LOAD EXPECTED CS2  
BIT #OR,T.CS2  
BEQ 7\$  
BIS #OR,E.CS2

3656	015372	023737	003500	003440	7\$:	CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3657	015400	001401				BEQ	8\$	:YES, CHECK CS2
3658	015402	104072				ERROR	72	:CS1 INCORRECT
3659	015404	023737	003510	003450	8\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3660	015412	001401				BEQ	9\$	:YES, CHECK ERROR REG.
3661	015414	104073				ERROR	73	:CS2 INCORRECT
3662	015416	023737	003514	003454	9\$:	CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
3663	015424	001401				BEQ	10\$	:CHECK BUS ADDRESS
3664	015426	104074				ERROR	74	:ERROR REG INCORRECT
3665	015430	023737	003504	003444	10\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3666	015436	001401				BEQ	11\$	:YES, CHECK WORD COUNT
3667	015440	104075				ERROR	75	:BUS ADDRESS INCORRECT
3668	015442	023737	003502	003442	11\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT REG.
3669	015450	001401				BEQ	12\$	:YES, CLEAR RK611
3670	015452	104076				ERROR	76	:WORD COUNT INCORRECT
3671	015454	012762	100000	000000	12\$:	MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
3672	015462	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3673	015470	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3674	015476	012737	000200	003500		MOV	#RDY,E.CS1	:LOAD EXPECTED CS1
3675	015504	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
3676	015512	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1
3677	015520	001401				BEQ	15\$	:YES, CHECK CS2
3678	015522	104077				ERROR	77	:CS1 INCORRECT
3679	015524	023737	003510	003450	15\$:	CMP	E.CS2,T.CS2	:CHECK IF NEM CLEARED
3680	015532	001401				BEQ	TST17	:YES, GO ON TO NEXT TEST
3681	015534	104100				ERROR	100	:CS2 INCORRECT

```

:*****
:TEST 17      BUS ADDRESS BIT 16
:
:  CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
:  IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
:  IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
:  SPECIFY A ONE WORD DATA TRANSFER FROM 200000.
:  READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
:  REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 177776.
:  CHECK BUS ADDRESS AND WORD COUNT.
:
:  NOTE:  THIS TEST IS ONLY EXECUTED IF MORE THAN 32K
:         OF MEMORY IS ON THE SYSTEM.
:*****

```

3698	015536	000004				TST17:	SCOPE	
3699	015540	012737	000144	001200		MOV	#100,\$TIMES	::DO 100. ITERATIONS
3700	015546	005737	043750			TST	\$KT11	:CHECK FOR MEMORY MANAGEMENT
3701	015552	100004				BPL	1\$	:NO, BYPASS TEST
3702	015554	022737	002000	044216		CMP	#2000,\$LSTBK	:CHECK IF ENOUGH MEMORY
3703	015562	103417				BLO	2\$	:YES, DO TEST
3704	015564				1\$:			
3705	015564	012737	000001	001200		MOV	#1,\$TIMES	:FORCE INTERATION COUNT TO 1
3706	015572	005227	177777			INC	#-1	:ONLY DO ONCE
3707	015576	001007				BNE	64\$	:NO, GO TO NEXT TEST
3708	015600	104401	052357			TYPE	,TSTBY1	:TYPE TEST N BYPASSED
3709	015604	013746	001220			MOV	\$TESTN,-(SP)	:SAVE \$TESTN FOR TYPEOUT
3710	015610	104402				TYPOC		:GO TYPE--OCTAL ASCII(ALL DIGITS)
3711	015612	104401	052367			TYPE	,TSTBY2	

```

3712 015616 000137 016652      64$: JMP      TST20      ;GO TO NEXT TEST
3713
3714 015622 013702 001270      2$:  MOV      $BASE,R2      ;LOAD K611 BASE
3715 015626 012737 002000 172354  MOV      #2000,KIPAR6      ;LOAD PAGE ADDRESS 6 FOR DATA
3716 015634 005237 177572      INC      SRO      ;TURN ON MEMORY MANAGEMENT
3717 015640 013737 064406 140000  MOV      NPRBUF,140000      ;LOAD WORD IN MEMORY
3718 015646 005037 177572      CLR      SRO      ;TURN OFF MEMORY MANAGEMENT
3719 015652 012737 015660 001110  MOV      #3$,$LPERR      ;LOAD LOOP ON ERROR LOCATION FOR
3720                                     ; SUBTEST LOOP
3721
3722 015660      3$:
3723 015660 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
3724 015666 012762 000040 000026  MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3725 015674 012762 177777 000002  MOV      #-1,RKWC(R2)
3726 015702 012737 000427 003500  MOV      #BA16!WRHEAD,E.CS1 ;LOAD COMMAND
3727 015710 012762 000427 000000  MOV      #BA16!WRHEAD,RKCS1(R2)
3728 015716 012700 000312      MOV      #50.*4+2,RO      ;ISSUE ENOUGH CLOCKS UNTIL
3729 015722 012762 000440 000026  5$:  MOV      #DMD!MCLK,RKMR1(R2) ; INDEX PULSE
3730 015730 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3731 015736 005300      DEC      RO
3732 015740 001370      BNE      5$
3733 015742 012700 000004      MOV      #4,RO      ;SIMULATE INDEX PULSE
3734 015746 012762 000240 000026  MOV      #DMD!MIND,RKMR1(R2)
3735 015754 012762 000640 000026  6$:  MOV      #DMD!MIND!MCLK,RKMR1(R2)
3736 015762 012762 000240 000026  MOV      #DMD!MIND,RKMR1(R2)
3737 015770 005300      DEC      RO
3738 015772 001370      BNE      6$
3739 015774 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3740 016002 012700 000045      MOV      #37.,RO      ;ISSUE 1 NPR TRANSFER
3741 016006 012762 000440 000026  7$:  MOV      #DMD!MCLK,RKMR1(R2)
3742 016014 012762 000040 000026  MOV      #DMD,RKMR1(R2)
3743 016022 005300      DEC      RO
3744 016024 001370      BNE      7$
3745 016026 016237 000000 003440  MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3746 016034 016237 000004 003444  MOV      RKBA(R2),T.BA      ;STORE BUS ADDRESS
3747 016042 016237 000002 003442  MOV      RKWC(R2),T.WC      ;STORE WOR
3748 016050 012700 000024      MOV      #20.,RO      ;WAIT FOR OUTPUT READY
3749 016054 005300      8$:  DEC      RO
3750 016056 001376      BNE      8$
3751 016060 016237 000010 003450  MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3752 016066 012737 000300 003510  MOV      #IR!OR,E.CS2      ;LOAD EXPECTED CS2
3753 016074 012737 000002 003504  MOV      #2,E.BA      ;LOAD EXPECTED BUS ADDRESS
3754 016102 005037 003502      CLR      E.WC      ;LOAD EXPECTED WORD COUNT
3755 016106 023737 003500 003440  CMP      E.CS1,T.CS1      ;CHECK COMMAND STATUS REG. 1 CORRECT
3756 016114 001401      BEQ      10$      ;YES, CHECK CS2
3757 016116 104101      ERROR    101      ;CS1 INCORRECT
3758 016120 023737 003510 003450  10$:  CMP      E.CS2,T.CS2      ;CHECK IF CS2 CORRECT
3759 016126 001401      BEQ      11$      ;YES, CHECK BUS ADDRESS CORRECT
3760 016130 104102      ERROR    102      ;CS2 INCORRECT
3761 016132 023737 003504 003444  11$:  CMP      E.BA,T.BA      ;CHECK IF BUS ADD INCORRECT
3762 016140 001401      BEQ      12$      ;YES, CHECK WORD COUNT CORRECT
3763 016142 104103      ERROR    103      ;BUS ADDRESS INCORRECT
3764 016144 023737 003502 003442  12$:  CMP      E.WC,T.WC      ;CHECK IF WORD COUNT CORRECT
3765 016152 001401      BEQ      13$      ;YES, CHECK DATA BUFFER
3766 016154 104104      ERROR    104      ;WORD COUNT INCORRECT
3767 016156 016237 000024 003462  13$:  MOV      RKDB(R2),T.DB      ;READ DATA BUFFER
    
```

3768	016164	013737	064406	003522	MOV	NPRBUF,E.DB	:LOAD EXPECTED DATA BUFFER
3769	016172	023737	003522	003462	CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3770	016200	001401			BEQ	15\$	:YES, CHECK IF LOOP ON ERROR
3771	016202	104105			ERROR	105	:DATA INCORRECT
3772	016204	104415			15\$:	SCOP1	:CHECK IF LOOP ON ERROR
3773	016206	012737	001777	172354	MOV	#2000-1,KIPAR6	:LOAD PAGE ADDRESS 6 FOR DATA
3774	016214	005237	177572		INC	SRO	:TURN ON MEMORY MANAGEMENT
3775	016220	013737	064406	140076	MOV	NPRBUF,140076	:LOAD WORDS IN MEMORY
3776	016226	013737	064410	140100	MOV	NPRBUF+2,140100	
3777	016234	005037	177572		CLR	SRO	:TURN OFF MEMORY MANAGEMENT
3778	016240	012737	016246	001110	MOV	#20\$,\$LPERR	:LOAD LOOP ON ERROR LOCATION FOR
3779							: SUBTEST LOOP
3780							
3781	016246				20\$:		
3782	016246	012762	100000	000000	MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
3783	016254	012762	000040	000026	MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
3784	016262	012737	177777	003502	MOV	#-1,E.WC	:LOAD WORD COUNT REG.
3785	016270	012762	177776	000002	MOV	#-2,RKWC(R2)	
3786	016276	005037	003504		CLR	E.BA	:LOAD BUS ADDRESS
3787	016302	012762	177776	000004	MOV	#177776,RKBA(R2)	
3788	016310	012737	000427	003500	MOV	#BA16!WRHEAD,E.CS1	:LOAD COMMAND
3789	016316	012762	000027	000000	MOV	#WRHEAD,RKCS1(R2)	
3790	016324	012700	000312		MOV	#50.+4+2,RO	:ISSUE ENOUGH CLOCKS UNTIL
3791	016330	012762	000440	000026	21\$:	MOV	#DMD!MCLK,RKMR1(R2) ; INDEX PULSE
3792	016336	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3793	016344	005300			DEC	RO	
3794	016346	001370			BNE	21\$	
3795	016350	012700	000004		MOV	#4,RO	:SIMULATE INDEX PULSE
3796	016354	012762	000240	000026	MOV	#DMD!MIND,RKMR1(R2)	
3797	016362	012762	000640	000026	22\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)
3798	016370	012762	000240	000026	MOV	#DMD!MIND,RKMR1(R2)	
3799	016376	005300			DEC	RO	
3800	016400	001370			BNE	22\$	
3801	016402	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3802	016410	012701	000002		MOV	#2,R1	:ISSUE 2 NPR TRANSFERS
3803	016414	012700	000045		23\$:	MOV	#37.,RO
3804	016420	012762	000440	000026	24\$:	MOV	#DMD!MCLK,RKMR1(R2)
3805	016426	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3806	016434	005300			DEC	RO	
3807	016436	001370			BNE	24\$	
3808	016440	016237	000000	003440	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3809	016446	016237	000004	003444	MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG.
3810	016454	016237	000002	003442	MOV	RKWC(R2),T.WC	:STORE WORD COUNT
3811	016462	005737	003502		TST	E.WC	:CHECK IF FIRST WORD
3812	016466	001404			BEQ	26\$	:NO, GET CS2
3813	016470	012700	000024		MOV	#20.,RO	:WAIT FOR OUTPUT READY
3814	016474	005300			25\$:	DEC	RO
3815	016476	001376			BNE	25\$	
3816	016500	016237	000010	003450	26\$:	MOV	RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
3817	016506	012737	000300	003510	MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3818	016514	023737	003500	003440	CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG 1 CORRECT
3819	016522	001401			BEQ	28\$	:YES, CHECK CS2
3820	016524	104101			ERROR	101	:CS1 INCORRECT
3821	016526	023737	003510	003450	28\$:	CMP	E.CS2,T.CS2 ;CHECK COMMAND STATUS REG 2 CORRECT
3822	016534	001401			BEQ	29\$	:YES, CHECK BUS ADDRESS
3823	016536	104102			ERROR	102	:CS2 INCORRECT



```

3824 016540 023737 003504 003444 29$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
3825 016546 001401 BEQ 30$ ;YES, CHECK WORD COUNT
3826 016550 104103 ERROR 103 ;BUS ADDRESS INCORRECT
3827 016552 023737 003502 003442 30$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
3828 016560 001401 BEQ 31$ ;YES, GET TEXT WORD
3829 016562 104104 ERROR 104 ;WORD COUNT INCORRECT
3830 016564 062737 000002 003504 31$: ADD #2,E.BA ;INCREMENT BUS ADDRESS AND
3831 016572 005237 003502 INC E.WC ;WORD COUNT
3832 016576 005301 DEC R1 ;CHECK IF FINISHED
3833 016600 001305 BNE 23$ ;NO, GET SECOND WORD
3834 016602 012700 000002 MOV #2,R0 ;LOAD COUNT AND ADDRESS WORD
3835 016606 012703 064406 MOV #NPRBUF,R3 ;DATA COMPARE
3836 016612 016237 000024 003462 35$: MOV RKDB(R2),T.DB ;READ DATA BUFFER
3837 016620 012337 003522 MOV (R3)+,E.DB ;GET EXPECTED DATA
3838 016624 023737 003522 003462 CMP E.DB,T.DB ;CHECK IF CORRECT
3839 016632 001401 BEQ 36$ ;YES, CHECK IF FINISHED
3840 016634 104105 ERROR 105 ;DATA BUFFER INCORRECT
3841 016636 005300 36$: DEC R0 ;CHECK IF FINISHED
3842 016640 001364 BNE 35$ ;NO READ SECOND WORD
3843 016642 104415 SCOP1 ;CHECK IF LOOP ON ERROR
3844 016644 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
  
```

```

*****
:TEST 20 BUS ADDRESS BIT 17
:
: CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
: IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
: IN 26 SECTOR FORMAT, CYLINDER 0, HEADER 0, DRIVE 0.
: SPECIFY A ONE WORD DATA TRANSFER FROM 400000.
: READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
: REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 377776.
: CHECK BUS ADDRESS AND WORD COUNT.
:
: NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 64K
: OF MEMORY IS ON THE SYSTEM.
  
```

```

*****
3860 TST20: SCOPE
3861 016652 000004 MOV #100, $TIMES ;DO 100. ITERATIONS
3862 016654 012737 000144 001200 TST $KT11 ;CHECK FOR MEMORY MANAGEMENT
3863 016662 005737 043750 BPL 1$ ;NO, BYPASS TEST
3864 016666 100004 CMP #4000, $LSTBK ;CHECK IF ENOUGH MEMORY
3865 016670 022737 004000 044216 BLO 2$ ;YES, DO TEST
3866 016676 103417
3867 016700 1$: MOV #1, $TIMES ;FORCE INTERATION COUNT TO 1
3868 016700 012737 000001 001200 INC #-1 ;ONLY DO ONCE
3869 016706 005227 177777 BNE 64$ ;NO, GO TO NEXT TEST
3870 016712 001007 TYPE ,TSTBY1 ;TYPE TEST N BYPASSED
3871 016714 104401 052357 MOV $TESTN, -(SP) ;SAVE $TESTN FOR TYPEOUT
3872 016720 013746 001220 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3873 016724 104402 TYPE ,TSTBY2
3874 016726 104401 052367 64$: JMP TST21 ;GO TO NEXT TEST
3875 016732 000137 017766
3876
3877 016736 013702 001270 2$: MOV $BASE, R2 ;LOAD K611 BASE
3878 016742 012737 004000 172354 MOV #4000, KIPAR6 ;LOAD PAGE ADDRESS 6 FOR DATA
3879 016750 005237 177572 INC SRO ;TURN ON MEMORY MANAGEMENT
  
```

3880	016754	013737	064406	140000	MOV	NPRBUF,140000	:LOAD WORD IN MEMORY
3881	016762	005037	177572		CLR	SRO	:TURN OFF MEMORY MANAGEMENT
3882	016766	012737	016774	001110	MOV	#3\$,SLPERR	:LOAD LOOP ON ERROR LOCATION FOR
3883							: SUBTEST LOOP
3884							
3885	016774				3\$:		
3886	016774	012762	100000	000000	MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
3887	017002	012762	000040	000026	MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
3888	017010	012762	177777	000002	MOV	#-1,RKWC(R2)	
3889	017016	012737	001027	003500	MOV	#BA17!WRHEAD,E.CS1	:LOAD COMMAND
3890	017024	012762	001027	000000	MOV	#BA17!WRHEAD,RKCS1(R2)	
3891	017032	012700	000312		MOV	#50.*4+2,RO	:ISSUE ENOUGH CLOCKS UNTIL
3892	017036	012762	000440	000026	5\$:	MOV	#DMD!MCLK,RKMR1(R2) ; INDEX PULSE
3893	017044	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3894	017052	005300			DEC	RO	
3895	017054	001370			BNE	5\$	
3896	017056	012700	000004		MOV	#4,RO	:SIMULATE INDEX PULSE
3897	017062	012762	000240	000026	MOV	#DMD!MIND,RKMR1(R2)	
3898	017070	012762	000640	000026	6\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)
3899	017076	012762	000240	000026	MOV	#DMD!MIND,RKMR1(R2)	
3900	017104	005300			DEC	RO	
3901	017106	001370			BNE	6\$	
3902	017110	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3903	017116	012700	000045		MOV	#37,RO	:ISSUE 1 NPR TRANSFER
3904	017122	012762	000440	000026	7\$:	MOV	#DMD!MCLK,RKMR1(R2)
3905	017130	012762	000040	000026	MOV	#DMD,RKMR1(R2)	
3906	017136	005300			DEC	RO	
3907	017140	001370			BNE	7\$	
3908	017142	016237	000000	003440	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3909	017150	016237	000004	003444	MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS
3910	017156	016237	000002	003442	MOV	RKWC(R2),T.WC	:STORE WOR
3911	017164	012700	000024		MOV	#20.,RO	:WAIT FOR OUTPUT READY
3912	017170	005300			8\$:	DEC	RO
3913	017172	001376			BNE	8\$	
3914	017174	016237	000010	003450	MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3915	017202	012737	000300	003510	MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3916	017210	012737	000002	003504	MOV	#2,E.BA	:LOAD EXPECTED BUS ADDRESS
3917	017216	005037	003502		CLR	E.WC	:LOAD EXPECTED WORD COUNT
3918	017222	023737	003500	003440	CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3919	017230	001401			BEQ	10\$	:YES, CHECK CS2
3920	017232	104101			ERROR	101	:CS1 INCORRECT
3921	017234	023737	003510	003450	10\$:	CMP	E.CS2,T.CS2
3922	017242	001401			BEQ	11\$	:YES, CHECK BUS ADDRESS CORRECT
3923	017244	104102			ERROR	102	:CS2 INCORRECT
3924	017246	023737	003504	003444	11\$:	CMP	E.BA,T.BA
3925	017254	001401			BEQ	12\$	:YES, CHECK WORD COUNT CORRECT
3926	017256	104103			ERROR	103	:BUS ADDRESS INCORRECT
3927	017260	023737	003502	003442	12\$:	CMP	E.WC,T.WC
3928	017266	001401			BEQ	13\$	:YES, CHECK DATA BUFFER
3929	017270	104104			ERROR	104	:WORD COUNT INCORRECT
3930	017272	016237	000024	003462	13\$:	MOV	RKDB(R2),T.DB
3931	017300	013737	064406	003522	MOV	NPRBUF,E.DB	:LOAD EXPECTED DATA BUFFER
3932	017306	023737	003522	003462	CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3933	017314	001401			BEQ	15\$	:YES, CHECK IF LOOP ON ERROR
3934	017316	104105			ERROR	105	:DATA INCORRECT
3935	017320	104415			15\$:	SCOP1	:CHECK IF LOOP ON ERROR

```

3936 017322 012737 003777 172354      MOV      #4000-1,KIPAR6 ;LOAD PAGE ADDRESS 6 FOR DATA
3937 017330 005237 177572                INC      SRO           ;TURN ON MEMORY MANAGEMENT
3938 017334 013737 064406 140076      MOV      NPRBUF,140076 ;LOAD WORDS IN MEMORY
3939 017342 013737 064410 140100      MOV      NPRBUF+2,140100
3940 017350 005037 177572                CLR      SRO           ;TURN OFF MEMORY MANAGEMENT
3941 017354 012737 017362 001110      MOV      #20$, $LPERR ;LOAD LOOP ON ERROR LOCATION FOR
3942                                     ; SUBTEST LOOP
3943
3944 017362                20$:
3945 017362 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
3946 017370 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3947 017376 012737 177777 003502      MOV      #-1,E.WC      ;LOAD WORD COUNT REG.
3948 017404 012762 177776 000002      MOV      #-2,RKWC(R2)
3949 017412 005037 003504                CLR      E.BA         ;LOAD BUS ADDRESS
3950 017416 012762 177776 000004      MOV      #177776,RKBA(R2)
3951 017424 012737 001027 003500      MOV      #BA17!WRHEAD,E.CS1 ;LOAD COMMAND
3952 017432 012762 000427 000000      MOV      #BA16!WRHEAD,RKCS1(R2)
3953 017440 012700 000312                MOV      #50.*4+2,R0   ;ISSUE ENOUGH CLOCKS UNTIL
3954 017444 012762 000440 000026      21$: MOV      #DMD!MCLK,RKMR1(R2) ; INDEX PULSE
3955 017452 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3956 017460 005300                DEC      R0
3957 017462 001370                BNE     21$
3958 017464 012700 000004                MOV      #4,R0         ;SIMULATE INDEX PULSE
3959 017470 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
3960 017476 012762 000640 000026      22$: MOV      #DMD!MIND!MCLK,RKMR1(R2)
3961 017504 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
3962 017512 005300                DEC      R0
3963 017514 001370                BNE     22$
3964 017516 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3965 017524 012701 000002                MOV      #2,R1         ;ISSUE 2 NPR TRANSFERS
3966 017530 012700 000045                23$: MOV      #37.,R0
3967 017534 012762 000440 000026      24$: MOV      #DMD!MCLK,RKMR1(R2)
3968 017542 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3969 017550 005300                DEC      R0
3970 017552 001370                BNE     24$
3971 017554 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3972 017562 016237 000004 003444      MOV      RKBA(R2),T.BA  ;STORE BUS ADDRESS REG.
3973 017570 016237 000002 003442      MOV      RKWC(R2),T.WC ;STORE WORD COUNT
3974 017576 005737 003502                TST     E.WC          ;CHECK IF FIRST WORD
3975 017602 001404                BEQ     26$          ;NO, GET CS2
3976 017604 012700 000024                MOV      #20.,R0      ;WAIT FOR OUTPUT READY
3977 017610 005300                25$: DEC      R0
3978 017612 001376                BNE     25$
3979 017614 016237 000010 003450      26$: MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
3980 017622 012737 000300 003510      MOV      #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3981 017630 023737 003500 003440      CMP     E.CS1,T.CS1  ;CHECK COMMAND STATUS REG 1 CORRECT
3982 017636 001401                BEQ     28$          ;YES, CHECK CS2
3983 017640 104101                ERROR   101         ;CS1 INCORRECT
3984 017642 023737 003510 003450      28$: CMP     E.CS2,T.CS2 ;CHECK COMMAND STATUS REG 2 CORRECT
3985 017650 001401                BEQ     29$          ;YES, CHECK BUS ADDRESS
3986 017652 104102                ERROR   102         ;CS2 INCORRECT
3987 017654 023737 003504 003444      29$: CMP     E.BA,T.BA  ;CHECK BUS ADDRESS CORRECT
3988 017662 001401                BEQ     30$          ;YES, CHECK WORD COUNT
3989 017664 104103                ERROR   103         ;BUS ADDRESS INCORRECT
3990 017666 023737 003502 003442      30$: CMP     E.WC,T.WC  ;CHECK WORD COUNT CORRECT
3991 017674 001401                BEQ     31$          ;YES, GET TEXT WORD

```

```

3992 017676 104104          ERROR 104          :WORD COUNT INCORRECT
3993 017700 062737 000002 003504 31$: ADD #2,E.BA      :INCREMENT BUS ADDRESS AND
3994 017706 005237 003502          INC E.WC         :WORD COUNT
3995 017712 005301          DEC R1           :CHECK IF FINISHED
3996 017714 001305          BNE 23$         :NO, GET SECOND WORD
3997 017716 012700 000002          MOV #2,R0       :LOAD COUNT AND ADDRESS WORD
3998 017722 012703 064406          MOV #NPRBUF,R3  :DATA COMPARE
3999 017726 016237 000024 003462 35$: MOV RKDB(R2),T.DB :READ DATA BUFFER
4000 017734 012337 003522          MOV (R3)+,E.DB  :GET EXPECTED DATA
4001 017740 023737 003522 003462          CMP E.DB,T.DB   :CHECK IF CORRECT
4002 017746 001401          BEQ 36$         :YES, CHECK IF FINISHED
4003 017750 104105          ERROR 105       :DATA BUFFER INCORRECT
4004 017752 005300          DEC R0          :CHECK IF FINISHED
4005 017754 001364          BNE 35$         :NO READ SECOND WORD
4006 017756 104415          SCOP1          :CHECK IF LOOP ON ERROR
4007 017760 012762 100000 000000          MOV #CCLR,RKCS1(R2) :CLEAR RK611

```

```

4008
4009
4010 *****
4011 *TEST 21 ADDRESSING GREATER THAN 96K
4012 *
4013 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
4014 * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
4015 * IN 26 SECTOR FORMAT, CYLINDER 0, HEADER 0, DRIVE 0.
4016 * SPECIFY A ONE WORD DATA TRANSFER FROM 600000.
4017 * READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
4018 * REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 577776.
4019 * CHECK BUS ADDRESS AND WORD COUNT.
4020 *
4021 * NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 96K
4022 * OF MEMORY IS ON THE SYSTEM.
4023 *****

```

```

4024 017766 000004          TST21: SCOPE
4025 017770 012737 000144 001200          MOV #100,$TIMES  ;;DO 100. ITERATIONS
4026 017776 005737 043750          TST $KT11       :CHECK FOR MEMORY MANAGEMENT
4027 020002 100004          BPL 1$          :NO, BYPASS TEST
4028 020004 022737 006000 044216          CMP #6000,$LSTBK :CHECK IF ENOUGH MEMORY
4029 020012 103417          BLO 2$          :YES, DO TEST
4030 020014          1$:
4031 020014 012737 000001 001200          MOV #1,$TIMES   :FORCE INTERATION COUNT TO 1
4032 020022 005227 177777          INC #-1         :ONLY DO ONCE
4033 020026 001007          BNE 64$         :NO, GO TO NEXT TEST
4034 020030 104401 052357          TYPE ,TSTBY1    :TYPE TEST N BYPASSED
4035 020034 013746 001220          MOV $TESTN,-(SP) :SAVE $TESTN FOR TYPEOUT
4036 020040 104402          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4037 020042 104401 052367          TYPE ,TSTBY2
4038 020046 000137 021102 64$: JMP TST22       :GO TO NEXT TEST
4039
4040 020052 013702 001270          2$: MOV $BASE,R2   :LOAD K611 BASE
4041 020056 012737 006000 172354          MOV #6000,KIPAR6 :LOAD PAGE ADDRESS 6 FOR DATA
4042 020064 005237 177572          INC SR0         :TURN ON MEMORY MANAGEMENT
4043 020070 013737 064406 140000          MOV NPRBUF,140000 :LOAD WORD IN MEMORY
4044 020076 005037 177572          CLR SR0         :TURN OFF MEMORY MANAGEMENT
4045 020102 012737 020110 001110          MOV #3$,$LPERR  :LOAD LOOP ON ERROR LOCATION FOR
4046 : SUBTEST LOOP
4047

```



```

4104 020470 012737 020476 001110 MOV #20$,SLPERR ;LOAD LOOP ON ERROR LOCATION FOR
4105 ; SUBTEST LOOP
4106
4107 020476 20$:
4108 020476 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4109 020504 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4110 020512 012737 177777 003502 MOV #-1,E.WC ;LOAD WORD COUNT REG.
4111 020520 012762 177776 000002 MOV #-2,RKWC(R2)
4112 020526 005037 003504 CLR E.BA ;LOAD BUS ADDRESS
4113 020532 012762 177776 000004 MOV #177776,RKBA(R2)
4114 020540 012737 001427 003500 MOV #BA16!BA17!WRHEAD,E.CS1 ;LOAD COMMAND
4115 020546 012762 001027 000000 MOV #BA17!WRHEAD,RKCS1(R2)
4116 020554 012700 000312 MOV #50.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL
4117 020560 012762 000440 000026 21$: MOV #DMD!MCLK,RKMR1(R2) ; INDEX PULSE
4118 020566 012762 000040 000026 MOV #DMD,RKMR1(R2)
4119 020574 005300 DEC R0
4120 020576 001370 BNE 21$
4121 020600 012700 000004 MOV #4,R0 ;SIMULATE INDEX PULSE
4122 020604 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
4123 020612 012762 000640 000026 22$: MOV #DMD!MIND!MCLK,RKMR1(R2)
4124 020620 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
4125 020626 005300 DEC R0
4126 020630 001370 BNE 22$
4127 020632 012762 000040 000026 MOV #DMD,RKMR1(R2)
4128 020640 012701 000002 MOV #2,R1 ;ISSUE 2 NPR TRANSFERS
4129 020644 012700 000045 23$: MOV #37,R0
4130 020650 012762 000440 000026 24$: MOV #DMD!MCLK,RKMR1(R2)
4131 020656 012762 000040 000026 MOV #DMD,RKMR1(R2)
4132 020664 005300 DEC R0
4133 020666 001370 BNE 24$
4134 020670 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4135 020676 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS REG.
4136 020704 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT
4137 020712 005737 003502 TST E.WC ;CHECK IF FIRST WORD
4138 020716 001404 BEQ 26$ ;NO, GET CS2
4139 020720 012700 000024 MOV #20.,R0 ;WAIT FOR OUTPUT READY
4140 020724 005300 25$: DEC R0
4141 020726 001376 BNE 25$
4142 020730 016237 000010 003450 26$: MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
4143 020736 012737 000300 003510 MOV #1R!OR,E.CS2 ;LOAD EXPECTED CS2
4144 020744 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND STATUS REG 1 CORRECT
4145 020752 001401 BEQ 28$ ;YES, CHECK CS2
4146 020754 104101 ERROR 101 ;CS1 INCORRECT
4147 020756 023737 003510 003450 28$: CMP E.CS2,T.CS2 ;CHECK COMMAND STATUS REG 2 CORRECT
4148 020764 001401 BEQ 29$ ;YES, CHECK BUS ADDRESS
4149 020766 104102 ERROR 102 ;CS2 INCORRECT
4150 020770 023737 003504 003444 29$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
4151 020776 001401 BEQ 30$ ;YES, CHECK WORD COUNT
4152 021000 104103 ERROR 103 ;BUS ADDRESS INCORRECT
4153 021002 023737 003502 003442 30$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
4154 021010 001401 BEQ 31$ ;YES, GET TEXT WORD
4155 021012 104104 ERROR 104 ;WORD COUNT INCORRECT
4156 021014 062737 000002 003504 31$: ADD #2,E.BA ;INCREMENT BUS ADDRESS AND
4157 021022 005237 003502 INC E.WC ; WORD COUNT
4158 021026 005301 DEC R1 ;CHECK IF FINISHED
4159 021030 001305 BNE 23$ ;NO, GET SECOND WORD
  
```

4160	021032	012700	000002			MOV	#2,RO	:LOAD COUNT AND ADDRESS WORD
4161	021036	012703	064406			MOV	#NPRBUF,R3	: DATA COMPARE
4162	021042	016237	000024	003462	358:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER
4163	021050	012337	003522			MOV	(R3)+,E.DB	:GET EXPECTED DATA
4164	021054	023737	003522	003462		CMP	E.DB,T.DB	:CHECK IF CORRECT
4165	021062	001401				BEQ	368	:YES, CHECK IF FINISHED
4166	021064	104105				ERROR	105	:DATA BUFFER INCORRECT
4167	021066	005300			368:	DEC	RO	:CHECK IF FINISHED
4168	021070	001364				BNE	358	:NO READ SECOND WORD
4169	021072	104415				SCOP1		:CHECK IF LOOP ON ERROR
4170	021074	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611

```

4171
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215

```

\*\*\*\*\*  
 :TEST 22 SILO FILL IN 18 BIT MODE  
 :  
 : CLEAR RK611 WITH CONTROLLER CLEAR. PUT CONTROLLER  
 : IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06  
 : IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0,  
 : SPECIFY 66 WORD DATA TRANSFER. CLOCK  
 : ALL 66 WORD INTO THE SILO. CHECK THAT ALL 66  
 : WORDS ARE CORRECT (16 LEAST SIGNIFICANT BITS).  
 :  
 :\*\*\*\*\*

```

4184 021102 000004
4185 021104 012737 000144 001200
4186 021112 013702 001270
4187 021116 012762 100000 000000
4188 021124 012762 000040 000026
4189 021132 012762 177676 000002
4190 021140 012737 177677 003502
4191 021146 012762 064406 000004
4192 021154 012737 064410 003504
4193 021162 012737 010027 003500
4194 021170 012762 010027 000000
4195 021176 012700 000312
4196
4197 021202 012762 000440 000026 58:
4198 021210 012762 000040 000026
4199 021216 005300
4200 021220 001370
4201 021222 012700 000004
4202 021226 012762 000240 000026
4203 021234 012762 000640 000026 68:
4204 021242 012762 000240 000026
4205 021250 005300
4206 021252 001370
4207 021254 012762 000040 000026
4208 021262 012737 000300 003510
4209 021270 012701 000102
4210 021274 012700 000050 78:
4211 021300 012762 000440 000026 88:
4212 021306 012762 000040 000026
4213 021314 005300
4214 021316 001370
4215 021320 016237 000000 003440

```

TST22: SCOPE  
 MOV #100.,STIMES ;;DO 100. ITERATIONS  
 MOV \$BASE,R2 ;LOAD RK611 BASE  
 MOV #CCLR,RKCS1(R2) ;CLEAR RK611  
 MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE  
 MOV #-66.,RKWC(R2) ;LOAD WORD COUNT  
 MOV #-65.,E.WC  
 MOV #NPRBUF,RKBA(R2);LOAD BUS ADDRESS  
 MOV #NPRBUF+2,E.BA  
 MOV #WRHEAD!CFMT,E.CS1 ;LOAD EXPECTED CS1  
 MOV #WRHEAD!CFMT,RKCS1(R2) ;ISSUE COMMAND  
 MOV #50.\*4+2,RO ;ISSUE ENOUGH CLOCKS DATA  
 ; INDEX PULSE  
 58: MOV #DMD!MCLK,RKMR1(R2)  
 MOV #DMD,RKMR1(R2)  
 DEC RO  
 BNE 58  
 MOV #4,RO ;SIMULATE INDEX PULSE  
 MOV #DMD!MIND,RKMR1(R2)  
 68: MOV #DMD!MIND!MCLK,RKMR1(R2)  
 MOV #DMD!MIND,RKMR1(R2)  
 DEC RO  
 BNE 68  
 MOV #DMD,RKMR1(R2)  
 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2  
 MOV #66.,R1 ;ISSUE 66 NPR TRANSFERS  
 78: MOV #40.,RO  
 88: MOV #DMD!MCLK,RKMR1(R2)  
 MOV #DMD,RKMR1(R2)  
 DEC RO  
 BNE 88  
 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1

```

4216 021326 016237 000004 003444      MOV      RKBA(R2),T.BA      ;STORE BUS ADDRESS REG. 2
4217 021334 016237 000002 003442      MOV      RKWC(R2),T.WC      ;STORE WORD COUNT
4218 021342 022737 064410 003504      CMP      #NPRBUF+2,E.BA      ;CHECK IF FIRST WORD
4219 021350 001004      BNE      10$                ;NO, CONTINUE
4220 021352 012700 000024      MOV      #20.,RO            ;WAIT FOR OUTPUT READY
4221 021356 005300      9$:      DEC      RO
4222 021360 001376      BNE      9$
4223 021362 016237 000010 003450 10$:     MOV      RKCS2(R2),T.CS2    ;STORE COMMAND AND STATUS REG. 2
4224 021370 005737 003502      TST      E.WC                ;CHECK IF LAST WORD
4225 021374 001003      BNE      11$                ;NO, CONTINUE
4226 021376 012737 000200 003510      MOV      #OR,E.CS2          ;LOAD EXPECTED CS2
4227 021404 023737 003500 003440 11$:     CMP      E.CS1,T.CS1        ;CHECK CS1 CORRECT
4228 021412 001401      BEQ      12$                ;YES, CONTINUE
4229 021414 104121      ERROR   121                 ;CS1 INCORRECT
4230 021416 023737 003510 003450 12$:     CMP      E.CS2,T.CS2        ;CHECK CS2 CORRECT
4231 021424 001401      BEQ      13$                ;YES, CONTINUE
4232 021426 104122      ERROR   122                 ;CS2 INCORRECT
4233 021430 023737 003504 003444 13$:     CMP      E.BA,T.BA          ;CHECK BUS ADDRESS CORRECT
4234 021436 001401      BEQ      14$                ;YES, CONTINUE
4235 021440 104123      ERROR   123                 ;BUS ADDRESS INCORRECT
4236 021442 023737 003502 003442 14$:     CMP      E.WC,T.WC          ;CHECK WORD COUNT REG CORRECT
4237 021450 001401      BEQ      15$                ;YES, CONTINUE
4238 021452 104124      ERROR   124                 ;WORD COUNT INCORRECT
4239 021454 062737 000002 003504 15$:     ADD      #2,E.BA            ;INCREMENT BUS ADDRESS AND
4240 021462 005237 003502      INC      E.WC                ;WORD COUNT
4241 021466 005301      DEC      R1                  ;CHECK IF SILO FULL
4242 021470 001301      BNE      7$                  ;NO, GET NEXT WORD
4243 021472 012700 000120      MOV      #2+40.,RO          ;ISSUE CLOCKS FOR TWO NPR'S
4244 021476 012762 000440 000026 20$:     MOV      #DMD!MCLK,RKMR1(R2)
4245 021504 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4246 021512 005300      DEC      RO
4247 021514 001370      BNE      20$
4248 021516 162737 000002 003504      SUB      #2,E.BA            ;ADJUST BUS ADDRESS AND WORD COUNT
4249 021524 005037 003502      CLR      E.WC
4250 021530 016237 000000 003440      MOV      RKCS1(R2),T.CS1    ;STORE COMMAND AND STATUS REG. 1
4251 021536 016237 000010 003450      MOV      RKCS2(R2),T.CS2    ;STORE COMMAND AND STATUS REG. 2
4252 021544 016237 000004 003444      MOV      RKBA(R2),T.BA      ;STORE BUS ADDRESS
4253 021552 016237 000002 003442      MOV      RKWC(R2),T.WC      ;STORE WORD COUNT
4254 021560 023737 003500 003440      CMP      E.CS1,T.CS1        ;CHECK CS1 CORRECT
4255 021566 001401      BEQ      21$                ;YES, CONTINUE
4256 021570 104125      ERROR   125                 ;CS1 INCORRECT
4257 021572 023737 003510 003450 21$:     CMP      E.CS2,T.CS2        ;CHECK CS2 CORRECT
4258 021600 001401      BEQ      22$                ;YES, CONTINUE
4259 021602 104126      ERROR   126                 ;CS2 INCORRECT
4260 021604 023737 003504 003444 22$:     CMP      E.BA,T.BA          ;CHECK BUS ADDRESS INCORRECT
4261 021612 001401      BEQ      23$                ;YES, CONTINUE
4262 021614 104127      ERROR   127                 ;BUS ADDRESS INCORRECT
4263 021616 023737 003502 003442 23$:     CMP      E.WC,T.WC          ;CHECK WORD COUNT CORRECT
4264 021624 001401      BEQ      24$                ;YES, CONTINUE
4265 021626 104130      ERROR   130                 ;WORD COUNT INCORRECT
4266 021630 012703 064406      MOV      #NPRBUF,R3         ;LOAD BUFFER ADDRESS FOR COMPARE
4267 021634 012737 000300 003510      MOV      #IR!OR,E.CS2       ;LOAD EXPECTED CS2
4268 021642 012701 000102      MOV      #66.,R1            ;LOAD COUNT
4269 021646 005037 003624      CLR      WRDCNT              ;INITIALIZE WORD COUNT
4270 021652 016237 000024 003462 25$:     MOV      RKDB(R2),T.DB      ;GET DATA BUFFER
4271 021660 012337 003522      MOV      (R3)+,E.DB         ;GET EXPECTED DATA

```



```

4272 021664 023737 003522 003462      CMP      E.DB,T.DB      ;CHECK IF DATA CORRECT
4273 021672 001401                    BEQ      26$           ;YES, CONTINUE
4274 021674 104131                    ERROR   131           ;DATA READ INCORRECT
4275 021676 012700 000050            26$:  MOV      #40.,R0   ;SET STALL
4276 021702 005300                    27$:  DEC      R0       ;RUN STALL TO ZERO
4277 021704 001376                    BNE     27$
4278 021706 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
4279 021714 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
4280 021722 022701 000001                    CMP      #1,R1        ;CHECK IF LAST WORD
4281 021726 001003                    BNE     28$           ;NO, CONTINUE
4282 021730 012737 000100 003510      MOV      #IR,E.CS2     ;LOAD EXPECTED CS2
4283 021736 023737 003500 003440 29$:  CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
4284 021744 001401                    BEQ      29$           ;YES, CONTINUE
4285 021746 104132                    ERROR   132           ;CS1 INCORRECT
4286 021750 023737 003510 003450 29$:  CMP      E.CS2,T.CS2    ;CHECK CS2 CORRECT
4287 021756 001401                    BEQ      30$           ;YES, CONTINUE
4288 021760 104133                    ERROR   133           ;CS2 INCORRECT
4289 021762 005237 003624            30$:  INC      WRDCNT      ;INCREMENT WORD COUNT
4290 021766 005301                    DEC      R1            ;CHECK IF FINISHED
4291 021770 001330                    BNE     25$           ;NO, CONTINUE
    
```

.SBTTL \*\*MFM READ LOOPBACK TESTS

```

*****
*TEST 23      READ LOOPBACK (PART 1)
*****
*
*      CLEAR RK611 WITH A CONTROLLER CLEAR, PUT CONTROLLER
*      IN DIAGNOSTIC MODE.  ISSUE A READ HEADER TO AN RK06
*      IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
*      CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
*      SIMULATE SECTOR PULSE, 255 ZEROES, A
*      ONE, AND A HEADER CONSISTING OF THE THREE
*      FOLLOWING WORDS:
*
*              177777
*              000000
*              177777
*
*      MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
*      IS TRANSFERRED.  CHECK THE SILO FOR CORRECT CONTENTS.
    
```

```

4315 021772 000004                    TST23: SCOPE
4316 021774 012737 000144 001200      MOV      #100.,$TIMES  ;;DO 100. ITERATIONS
4317 022002 013702 001270                    MOV      $BASE,R2     ;LOAD RK611 BASE
4318 022006 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
4319 022014 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4320 022022 012762 000025 000000      MOV      #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
4321 022030 012700 000312                    MOV      #50.*4+2,R0   ;ISSUE ENOUGH CLOCKS UNTIL READY
4322 022034 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4323 022042 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4324 022050 005300                    DEC      R0
4325 022052 001370                    BNE     1$
4326 022054 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4327 022062 012762 000040 000026      MOV      #DMD,RKMR1(R2)
    
```

4328	022070	005037	003614			CLR	PR.BIT		;INITIALIZE PRESENT BIT AND
4329	022074	005037	003616			CLR	M1.BIT		; PREVIOUS BIT
4330	022100	012700	000341			MOV	#225.,R0		
4331	022104	004737	043540		2\$:	JSR	PC,RDBIT		;SIMULATE SYNCH
4332	022110	005300				DEC	R0		
4333	022112	001374				BNE	2\$		
4334	022114	012737	000001	003614		MOV	#1,PR.BIT		
4335	022122	004737	043540			JSR	PC,RDBIT		
4336	022126	012701	000003			MOV	#3,R1		;LOAD NUMBER OF WORDS
4337	022132	012703	064326			MOV	#HEAD1,R3		;LOAD ADDRESS OF DATA
4338	022136	012737	000025	003500		MOV	#RDHEAD,E.CS1		;LOAD EXPECTED CS1
4339	022144	012304			5\$:	MOV	(R3)+,R4		;GET DATA
4340	022146	012700	000020			MOV	#16.,R0		;LOAD BIT COUNT
4341	022152	013737	003614	003616	6\$:	MOV	PR.BIT,M1.BIT		;STORE PREVIOUS BIT
4342	022160	006004				ROR	R4		;GET NEXT BIT
4343	022162	103403				BCS	7\$		;CHECK IF 1
4344	022164	005037	003614			CLR	PR.BIT		;NO, ZERO
4345	022170	000403				BR	8\$		;SIMULATE READ DATA
4346									
4347	022172	012737	000001	003614	7\$:	MOV	#1,PR.BIT		;ONE
4348	022200	004737	043540		8\$:	JSR	PC,RDBIT		;SIMULATE READ DATA
4349	022204	016237	000000	003440		MOV	RKCS1(R2),T.CS1		;READ COMMAND AND STATUS REG. 1
4350	022212	023737	003500	003440		CMP	E.CS1,T.CS1		;CHECK IF CS1 CORRECT
4351	022220	001417				BEQ	9\$		;YES, SIMULATE NEXT BIT
4352	022222	012737	000003	003624		MOV	#3,WRDCNT		;LOAD WORD COUNT
4353	022230	160137	003624			SUB	R1,WRDCNT		
4354	022234	012737	000020	003622		MOV	#16.,BITCNT		;LOAD BIT COUNT
4355	022242	160037	003622			SUB	R0,BITCNT		
4356	022246	104150				ERROR	150		;CS1 INCORRECT DURING HEADER
4357	022250	012762	100000	000000		MOV	#CCLR,RKCS1(R2)		;CLEAR RK611
4358	022256	000522				BR	TST24		;GO ON TO NEXT TEST
4359									
4360	022260	005300			9\$:	DEC	R0		;CHECK IF READY FOR NEXT WORD
4361	022262	001333				BNE	6\$		;NO, GET NEXT BIT
4362	022264	005301				DEC	R1		;CHECK IF HEADER FINISHED
4363	022266	001326				BNE	5\$		;NO, GET NEXT WORD
4364	022270	012700	000004			MOV	#4,R0		;LOAD COUNT FOR POSTAMBLE
4365	022274	013737	003614	003616	15\$:	MOV	PR.BIT,M1.BIT		;STORE LAST BIT
4366	022302	005037	003614			CLR	PR.BIT		;LOAD NEXT BIT
4367	022306	004737	043540			JSR	PC,RDBIT		;SIMULATE 1 BIT READ
4368	022312	005300				DEC	R0		;CHECK IF TIME FOR READY
4369	022314	001367				BNE	15\$		;NO, CONTINUE WITH POSTAMBLE
4370	022316	016237	000000	003440		MOV	RKCS1(R2),T.CS1		;GET CURRENT CS1
4371	022324	016237	000010	003450		MOV	RKCS2(R2),T.CS2		;GET CURRENT CS2
4372	022332	012737	000224	003500		MOV	#RDY!RDHEAD<^C<GO>>,E.CS1		;LOAD EXPECTED CS1
4373	022340	012737	000300	003510		MOV	#OR!IR,E.CS2		;LOAD EXPECTED CS2
4374	022346	023737	003500	003440		CMP	E.CS1,T.CS1		;CHECK CS1 CORRECT
4375	022354	001401				BEQ	16\$		;YES, CHECK CS2
4376	022356	104151				ERROR	151		;CS1 INCORRECT
4377	022360	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2		;CHECK CS2 CORRECT
4378	022366	001401				BEQ	17\$		;YES, CHECK DATA
4379	022370	104152				ERROR	152		;CS2 INCORRECT
4380	022372	005037	003624		17\$:	CLR	WRDCNT		;INITIALIZE WORD COUNT
4381	022376	012703	064326			MOV	#HEAD1,R3		;GET ADDRESS OF DATA
4382	022402	012337	003522		20\$:	MOV	(R3)+,E.DB		;GET EXPECTED DATA
4383	022406	016237	000024	003462		MOV	RKDB(R2),T.DB		;GET ACTUAL DATA

```

4384 022414 016237 000000 003440      MOV      RKCS1(R2),T.CS1  ;STORE COMMAND AND STATUS REG. 1
4385 022422 016237 000010 003450      MOV      RKCS2(R2),T.CS2  ;STORE COMMAND AND STATUS REG. 2
4386 022430 022737 000002 003624      CMP      #2,WRDCNT        ;CHECK IF LAST WORD IN DATA BUFFER
4387 022436 001003                BNE      21$              ;NO, CHECK CS1
4388 022440 012737 000100 003510      MOV      #IR,E.CS2        ;STORE EXPECTED CS2
4389 022446 023737 003500 003440 21$:      CMP      E.CS1,T.CS1      ;CHECK CS1 CORRECT
4390 022454 001402                BEQ      22$              ;YES, CHECK CS2
4391 022456 104153                ERROR    153              ;CS1 INCORRECT
4392 022460 000421                BR       TST24            ;;GO ON TO NEXT TEST
4393
4394 022462 023737 003510 003450 22$:      CMP      E.CS2,T.CS2      ;CHECK CS2 CORRECT
4395 022470 001402                BEQ      23$              ;YES, CHECK DATA
4396 022472 104154                ERROR    154              ;CS2 INCORRECT
4397 022474 000413                BR       TST24            ;;GO ON TO NEXT TEST
4398
4399 022476 023737 003522 003462 23$:      CMP      E.DB,T.DB        ;CHECK IF DATA CORRECT
4400 022504 001401                BEQ      24$              ;YES, GET NEXT HEADER WORD
4401 022506 104155                ERROR    155              ;DATA INCORRECT
4402 022510 005237 003624                24$:      INC      WRDCNT          ;INCREMENT WORD COUNT
4403 022514 022737 000003 003624      CMP      #3,WRDCNT        ;CHECK IF ALL THREE WORDS CHECK
4404 022522 001327                BNE      20$              ;NO, GET NEXT WORD
4405

```

```

*****
*TEST 24      READ LOOPBACK (PART 2)
*****

```

```

*
* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
* SIMULATE SECTOR PULSE, 255 ZEROES, A ONE,
* AND A HEADER CONSISTING OF THE THREE
* FOLLOWING WORDS:

```

```

*
*      000000
*      177777
*      000000

```

```

*
* MAKE SURE THAT READY COMES UP AFTER THIRD WORD
* IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

```

```

*****
TST24: SCOPE
*****

```

```

4425 022524 000004                MOV      #100,STIMES      ;;DO 100. ITERATIONS
4426 022526 012737 000144 001200      MOV      $BASE,R2        ;LOAD RK611 BASE
4427 022534 013702 001270                MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
4428 022540 012762 100000 000000      MOV      #DMD,RKMR1(R2)  ;PUT RK611 IN DIAGNOSTIC MODE
4429 022546 012762 000040 000026      MOV      #RDHEAD,RKCS1(R2);ISSUE READ HEADER
4430 022554 012762 000025 000000      MOV      #50,*4+2,RO      ;ISSUE ENOUGH CLOCKS UNTIL READY
4431 022562 012700 000312                1$:      MOV      #DMD!MCLK,RKMR1(R2); FOR SECTOR PULSE
4432 022566 012762 000440 000026      MOV      #DMD,RKMR1(R2)
4433 022574 012762 000040 000026      DEC      RO
4434 022602 005300                BNE      1$
4435 022604 001370                MOV      #DMD!MSP,RKMR1(R2);SIMULATE SECTOR PULSE
4436 022606 012762 000140 000026      MOV      #DMD,RKMR1(R2)
4437 022614 012762 000040 000026      CLR      PR.BIT          ;INITIALIZE PRESENT BIT AND
4438 022622 005037 003614                CLR      M1.BIT          ; PREVIOUS BIT
4439 022626 005037 003616

```

4440	022632	012700	000341			MOV	#225.,R0	
4441	022636	004737	043540		2\$:	JSR	PC,RDBIT	:SIMULATE SYNCH
4442	022642	005300				DEC	R0	
4443	022644	001374				BNE	2\$	
4444	022646	012737	000001	003614		MOV	#1,PR.BIT	
4445	022654	004737	043540			JSR	PC,RDBIT	
4446	022660	012701	000003			MOV	#3,R1	:LOAD NUMBER OF WORDS
4447	022664	012703	064334			MOV	#HEAD2,R3	:LOAD ADDRESS OF DATA
4448	022670	012737	000025	003500		MOV	#RDHEAD,E.CS1	:LOAD EXPECTED CS1
4449	022676	012304			5\$:	MOV	(R3)+,R4	:GET DATA
4450	022700	012700	000020			MOV	#16.,R0	:LOAD BIT COUNT
4451	022704	013737	003614	003616	6\$:	MOV	PR.BIT,M1.BIT	:STORE PREVIOUS BIT
4452	022712	006004				ROR	R4	:GET NEXT BIT
4453	022714	103403				BCS	7\$	:CHECK IF 1
4454	022716	005037	003614			CLR	PR.BIT	:NO, ZERO
4455	022722	000403				BR	8\$	:SIMULATE READ DATA
4456								
4457	022724	012737	000001	003614	7\$:	MOV	#1,PR.BIT	:ONE
4458	022732	004737	043540		8\$:	JSR	PC,RDBIT	:SIMULATE READ DATA
4459	022736	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:READ COMMAND AND STATUS REG. 1
4460	022744	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK IF CS1 CORRECT
4461	022752	001417				BEQ	9\$	:YES, SIMULATE NEXT BIT
4462	022754	012737	000003	003624		MOV	#3,WRDCNT	:LOAD WORD COUNT
4463	022762	160137	003624			SUB	R1,WRDCNT	
4464	022766	012737	000020	003622		MOV	#16.,BITCNT	:LOAD BIT COUNT
4465	022774	160037	003622			SUB	R0,BITCNT	
4466	023000	104150				ERROR	150	:CS1 INCORRECT DURING HEADER
4467	023002	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611
4468	023010	000522				BR	TST25	:GO ON TO NEXT TEST
4469								
4470	023012	005300			9\$:	DEC	R0	:CHECK IF READY FOR NEXT WORD
4471	023014	001333				BNE	6\$	:NO, GET NEXT BIT
4472	023016	005301				DEC	R1	:CHECK IF HEADER FINISHED
4473	023020	001326				BNE	5\$	:NO, GET NEXT WORD
4474	023022	012700	000004			MOV	#4,R0	:LOAD COUNT FOR POSTAMBLE
4475	023026	013737	003614	003616	15\$:	MOV	PR.BIT,M1.BIT	:STORE LAST BIT
4476	023034	005037	003614			CLR	PR.BIT	:LOAD NEXT BIT
4477	023040	004737	043540			JSR	PC,RDBIT	:SIMULATE 1 BIT READ
4478	023044	005300				DEC	R0	:CHECK IF TIME FOR READY
4479	023046	001367				BNE	15\$	:NO, CONTINUE WITH POSTAMBLE
4480	023050	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:GET CURRENT CS1
4481	023056	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:GET CURRENT CS2
4482	023064	012737	000224	003500		MOV	#RDY!RDHEAD<^C<GO>>,E.CS1	:LOAD EXPECTED CS1
4483	023072	012737	000300	003510		MOV	#OR!IR,E.CS2	:LOAD EXPECTED CS2
4484	023100	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4485	023106	001401				BEQ	16\$	:YES, CHECK CS2
4486	023110	104151				ERROR	151	:CS1 INCORRECT
4487	023112	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
4488	023120	001401				BEQ	17\$	:YES, CHECK DATA
4489	023122	104152				ERROR	152	:CS2 INCORRECT
4490	023124	005037	003624		17\$:	CLR	WRDCNT	:INITIALIZE WORD COUNT
4491	023130	012703	064334			MOV	#HEAD2,R3	:GET ADDRESS OF DATA
4492	023134	012337	003522		20\$:	MOV	(R3)+,E.DB	:GET EXPECTED DATA
4493	023140	016237	000024	003462		MOV	RKDB(R2),T.DB	:GET ACTUAL DATA
4494	023146	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
4495	023154	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2

```

4496 023162 022737 000002 003624      CMP      #2,WRDCNT      ;CHECK IF LAST WORD IN DATA BUFFER
4497 023170 001003                    BNE      21$          ;NO, CHECK CS1
4498 023172 012737 000100 003510      MOV      #IR,E.CS2    ;STORE EXPECTED CS2
4499 023200 023737 003500 003440 21$:  CMP      E.CS1,T.CS1  ;CHECK CS1 CORRECT
4500 023206 001402                    BEQ      22$          ;YES, CHECK CS2
4501 023210 104153                    ERROR    153          ;CS1 INCORRECT
4502 023212 000421                    BR       TST25        ;;GO ON TO NEXT TEST
4503
4504 023214 023737 003510 003450 22$:  CMP      E.CS2,T.CS2  ;CHECK CS2 CORRECT
4505 023222 001402                    BEQ      23$          ;YES, CHECK DATA
4506 023224 104154                    ERROR    154          ;CS2 INCORRECT
4507 023226 000413                    BR       TST25        ;;GO ON TO NEXT TEST
4508
4509 023230 023737 003522 003462 23$:  CMP      E.DB,T.DB    ;CHECK IF DATA CORRECT
4510 023236 001401                    BEQ      24$          ;YES, GET NEXT HEADER WORD
4511 023240 104155                    ERROR    155          ;DATA INCORRECT
4512 023242 005237 003624                    INC      WRDCNT        ;INCREMENT WORD COUNT
4513 023246 022737 000003 003624 24$:  CMP      #3,WRDCNT    ;CHECK IF ALL THREE WORDS CHECK
4514 023254 001327                    BNE      20$          ;NO, GET NEXT WORD
4515

```

\*\*\*\*\*  
 : \*TEST 25 READ LOOPBACK (PART 3)  
 : \*

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
 IN DIAGNOSTIC MOVE. ISSUE A READ HEADER TO AN RK06  
 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
 CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES  
 SIMULATE SECTOR PULSE, 255 ZEROES, A  
 ONE, AND A HEADER CONSISTING OF THE THREE  
 FOLLOWING WORDS:

125252  
 052525  
 125252

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD  
 IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

\*\*\*\*\*  
 : \*TST25: SCOPE  
 : \*

```

4535 023256 000004                    TST25:  MOV      #100, $TIMES    ;;DO 100. ITERATIONS
4536 023260 012737 000144 001200      MOV      $BASE,R2     ;LOAD RK611 BASE
4537 023266 013702 001270                    MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
4538 023272 012762 100000 000000      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4539 023300 012762 000040 000026      MOV      #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
4540 023306 012762 000025 000000      MOV      #50.*4+2,RO  ;ISSUE ENOUGH CLOCKS UNTIL READY
4541 023314 012700 000312                    MOV      #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4542 023320 012762 000440 000026 1$:  MOV      #DMD,RKMR1(R2)
4543 023326 012762 000040 000026      MOV      RO
4544 023334 005300                    DEC      1$
4545 023336 001370                    BNE      1$
4546 023340 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4547 023346 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4548 023354 005037 003614                    CLR      PR.BIT        ;INITIALIZE PRESENT BIT AND
4549 023360 005037 003614                    CLR      M1.BIT        ; PREVIOUS BIT
4550 023364 012700 000341                    MOV      #225.,RO
4551 023370 004737 043540 2$:  JSR      PC,RDBIT      ;SIMULATE SYNCH

```

4552	023374	005300			DEC	RO		
4553	023376	001374			BNE	2\$		
4554	023400	012737	000001	003614	MOV	#1,PR.BIT		
4555	023406	004737	043540		JSR	PC,RDBIT		
4556	023412	012701	000003		MOV	#3,R1	:LOAD NUMBER OF WORDS	
4557	023416	012703	064342		MOV	#HEAD3,R3	:LOAD ADDRESS OF DATA	
4558	023422	012737	000025	003500	MOV	#RDHEAD,E.CS1	:LOAD EXPECTED CS1	
4559	023430	012304			5\$:	MOV	(R3)+,R4	:GET DATA
4560	023432	012700	000020		MOV	#16.,RO	:LOAD BIT COUNT	
4561	023436	013737	003614	003616	6\$:	MOV	PR.BIT,M1.BIT	:STORE PREVIOUS BIT
4562	023444	006004			ROR	R4	:GET NEXT BIT	
4563	023446	103403			BCS	7\$	:CHECK IF 1	
4564	023450	005037	003614		CLR	PR.BIT	:NO, ZERO	
4565	023454	000403			BR	8\$	:SIMULATE READ DATA	
4566								
4567	023456	012737	000001	003614	7\$:	MOV	#1,PR.BIT	:ONE
4568	023464	004737	043540		8\$:	JSR	PC,RDBIT	:SIMULATE READ DATA
4569	023470	016237	000000	003440	MOV	RKCS1(R2),T.CS1	:READ COMMAND AND STATUS REG. 1	
4570	023476	023737	003500	003440	CMP	E.CS1,T.CS1	:CHECK IF CS1 CORRECT	
4571	023504	001417			BEQ	9\$	:YES, SIMULATE NEXT BIT	
4572	023506	012737	000003	003624	MOV	#3,WRDCNT	:LOAD WORD COUNT	
4573	023514	160137	003624		SUB	R1,WRDCNT		
4574	023520	012737	000020	003622	MOV	#16.,BITCNT	:LOAD BIT COUNT	
4575	023526	160037	003622		SUB	RO,BITCNT		
4576	023532	104150			ERROR	150	:CS1 INCORRECT DURING HEADER	
4577	023534	012762	100000	000000	MOV	#CCLR,RKCS1(R2)	:CLEAR RK611	
4578	023542	000522			BR	TST26	:GO ON TO NEXT TEST	
4579								
4580	023544	005300			9\$:	DEC	RO	:CHECK IF READY FOR NEXT WORD
4581	023546	001333			BNE	6\$	:NO, GET NEXT BIT	
4582	023550	005301			DEC	R1	:CHECK IF HEADER FINISHED	
4583	023552	001326			BNE	5\$	:NO, GET NEXT WORD	
4584	023554	012700	000004		MOV	#4,RO	:LOAD COUNT FOR POSTAMBLE	
4585	023560	013737	003614	003616	15\$:	MOV	PR.BIT,M1.BIT	:STORE LAST BIT
4586	023566	005037	003614		CLR	PR.BIT	:LOAD NEXT BIT	
4587	023572	004737	043540		JSR	PC,RDBIT	:SIMULATE 1 BIT READ	
4588	023576	005300			DEC	RO	:CHECK IF TIME FOR READY	
4589	023600	001367			BNE	15\$	:NO, CONTINUE WITH POSTAMBLE	
4590	023602	016237	000000	003440	MOV	RKCS1(R2),T.CS1	:GET CURRENT CS1	
4591	023610	016237	000010	003450	MOV	RKCS2(R2),T.CS2	:GET CURRENT CS2	
4592	023616	012737	000224	003500	MOV	#RDY!RDHEAD<^C<GO>>,E.CS1	:LOAD EXPECTED CS1	
4593	023624	012737	000300	003510	MOV	#OR!IR,E.CS2	:LOAD EXPECTED CS2	
4594	023632	023737	003500	003440	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT	
4595	023640	001401			BEQ	16\$	:YES, CHECK CS2	
4596	023642	104151			ERROR	151	:CS1 INCORRECT	
4597	023644	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
4598	023652	001401			BEQ	17\$	:YES, CHECK DATA	
4599	023654	104152			ERROR	152	:CS2 INCORRECT	
4600	023656	005037	003624		17\$:	CLR	WRDCNT	:INITIALIZE WORD COUNT
4601	023662	012703	064342		MOV	#HEAD3,R3	:GET ADDRESS OF DATA	
4602	023666	012337	003522		20\$:	MOV	(R3)+,E.DB	:GET EXPECTED DATA
4603	023672	016237	000024	003462	MOV	RKDB(R2),T.DB	:GET ACTUAL DATA	
4604	023700	016237	000000	003440	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1	
4605	023706	016237	000010	003450	MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2	
4606	023714	022737	000002	003624	CMP	#2,WRDCNT	:CHECK IF LAST WORD IN DATA BUFFER	
4607	023722	001003			BNE	21\$	:NO, CHECK CS1	

```

4608 023724 012737 000100 003510      MOV      #IR,E.CS2      ;STORE EXPECTED CS2
4609 023732 023737 003500 003440 21$:  CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
4610 023740 001402          BEQ      22$            ;YES, CHECK CS2
4611 023742 104153          ERROR   153            ;CS1 INCORRECT
4612 023744 000421          BR       TST26         ;;GO ON TO NEXT TEST
4613
4614 023746 023737 003510 003450 22$:  CMP      E.CS2,T.CS2    ;CHECK CS2 CORRECT
4615 023754 001402          BEQ      23$            ;YES, CHECK DATA
4616 023756 104154          ERROR   154            ;CS2 INCORRECT
4617 023760 000413          BR       TST26         ;;GO ON TO NEXT TEST
4618
4619 023762 023737 003522 003462 23$:  CMP      E.DB,T.DB      ;CHECK IF DATA CORRECT
4620 023770 001401          BEQ      24$            ;YES, GET NEXT HEADER WORD
4621 023772 104155          ERROR   155            ;DATA INCORRECT
4622 023774 005237 003624          INC      WRDCNT         ;INCREMENT WORD COUNT
4623 024000 022737 000003 003624 24$:  CMP      #3,WRDCNT     ;CHECK IF ALL THREE WORDS CHECK
4624 024006 001327          BNE     20$            ;NO, GET NEXT WORD
4625
  
```

```

*****
*TEST 26      READ LOOPBACK (PART 4)
  
```

```

*
* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
* IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
* SIMULATE SECTOR PULSE, 225 ZEROES, A
* ONE, AND A HEADER CONSISTING OF THE THREE
* FOLLOWING WORDS:
  
```

```

044444
022222
111111
  
```

```

* MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
* IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
  
```

```

*****
TST26: SCOPE
4645 024010 000004          MOV      #100,$TIMES    ;;DO 100. ITERATIONS
4646 024012 012737 000144 001200  MOV      $BASE,R2      ;LOAD RK611 BASE
4647 024020 013702 001270          MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
4648 024024 012762 100000 000000  MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4649 024032 012762 000040 000026  MOV      #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
4650 024040 012762 000025 000000  MOV      #50.*4+2,R0    ;ISSUE ENOUGH CLOCKS UNTIL READY
4651 024046 012700 000312          MOV      #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4652 024052 012762 000440 000026 1$:  MOV      #DMD,RKMR1(R2)
4653 024060 012762 000040 000026  DEC      R0
4654 024066 005300          BNE     1$
4655 024070 001370          MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4656 024072 012762 000140 000026  MOV      #DMD,RKMR1(R2)
4657 024100 012762 000040 000026  CLR      PR.BIT        ;INITIALIZE PRESENT BIT AND
4658 024106 005037 003614          CLR      M1.BIT        ; PREVIOUS BIT
4659 024112 005037 003616          MOV      #225.,R0
4660 024116 012700 000341          JSR     PC,RDBIT       ;SIMULATE SYNCH
4661 024122 004737 043540 2$:  DEC      R0
4662 024126 005300          BNE     2$
4663 024130 001374
  
```

```

4664 024132 012737 000001 003614 MOV #1,PR.BIT
4665 024140 004737 043540 JSR PC,RDBIT
4666 024144 012701 000003 MOV #3,R1 ;LOAD NUMBER OF WORDS
4667 024150 012703 064356 MOV #HEAD4,R3 ;LOAD ADDRESS OF DATA
4668 024154 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
4669 024162 012304 5$: MOV (R3)+,R4 ;GET DATA
4670 024164 012700 000020 MOV #16,,R0 ;LOAD BIT COUNT
4671 024170 013737 003614 003616 6$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4672 024176 006004 ROR R4 ;GET NEXT BIT
4673 024200 103403 BCS 7$ ;CHECK IF 1
4674 024202 005037 003614 CLR PR.BIT ;NO, ZERO
4675 024206 000403 BR 8$ ;SIMULATE READ DATA
4676
4677 024210 012737 000001 003614 7$: MOV #1,PR.BIT ;ONE
4678 024216 004737 043540 8$: JSR PC,RDBIT ;SIMULATE READ DATA
4679 024222 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;READ COMMAND AND STATUS REG. 1
4680 024230 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
4681 024236 001417 BEQ 9$ ;YES, SIMULATE NEXT BIT
4682 024240 012737 000003 003624 MOV #3,WRDCNT ;LOAD WORD COUNT
4683 024246 160137 003624 SUB R1,WRDCNT
4684 024252 012737 000020 003622 MOV #16,,BITCNT ;LOAD BIT COUNT
4685 024260 160037 003622 SUB R0,BITCNT
4686 024264 104150 ERROR 150 ;CS1 INCORRECT DURING HEADER
4687 024266 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4688 024274 000522 BR TST27 ;;GO ON TO NEXT TEST
4689
4690 024276 005300 9$: DEC R0 ;CHECK IF READY FOR NEXT WORD
4691 024300 001333 BNE 6$ ;NO, GET NEXT BIT
4692 024302 005301 DEC R1 ;CHECK IF HEADER FINISHED
4693 024304 001326 BNE 5$ ;NO, GET NEXT WORD
4694 024306 012700 000004 MOV #4,R0 ;LOAD COUNT FOR POSTAMBLE
4695 024312 013737 003614 003616 15$: MOV PR.BIT,M1.BIT ;STORE LAST BIT
4696 024320 005037 003614 CLR PR.BIT ;LOAD NEXT BIT
4697 024324 004737 043540 JSR PC,RDBIT ;SIMULATE 1 BIT READ
4698 024330 005300 DEC R0 ;CHECK IF TIME FOR READY
4699 024332 001367 BNE 15$ ;NO, CONTINUE WITH POSTAMBLE
4700 024334 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET CURRENT CS1
4701 024342 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;GET CURRENT CS2
4702 024350 012737 000224 003500 MOV #RDY!RDHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4703 024356 012737 000300 003510 MOV #OR!IR,E.CS2 ;LOAD EXPECTED CS2
4704 024364 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4705 024372 001401 BEQ 16$ ;YES, CHECK CS2
4706 024374 104151 ERROR 151 ;CS1 INCORRECT
4707 024376 023737 003510 003450 16$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4708 024404 001401 BEQ 17$ ;YES, CHECK DATA
4709 024406 104152 ERROR 152 ;CS2 INCORRECT
4710 024410 005037 003624 17$: CLR WRDCNT ;INITIALIZE WORD COUNT
4711 024414 012703 064356 MOV #HEAD4,R3 ;GET ADDRESS OF DATA
4712 024420 012337 003522 20$: MOV (R3)+,E.DB ;GET EXPECTED DATA
4713 024424 016237 000024 003462 MOV RKDB(R2),T.DB ;GET ACTUAL DATA
4714 024432 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4715 024440 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4716 024446 022737 000002 003624 CMP #2,WRDCNT ;CHECK IF LAST WORD IN DATA BUFFER
4717 024454 001003 BNE 21$ ;NO, CHECK CS1
4718 024456 012737 000100 003510 MOV #IR,E.CS2 ;STORE EXPECTED CS2
4719 024464 023737 003500 003440 21$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
  
```



```

4720 024472 001402      BEQ      22$      :YES, CHECK CS2
4721 024474 104153      ERROR    153      :CS1 INCORRECT
4722 024476 000421      BR       TST27    ;;GO ON TO NEXT TEST
4723
4724 024500 023737 003510 003450 22$:  CMP      E.CS2,T.CS2 :CHECK CS2 CORRECT
4725 024506 001402      BEQ      23$      :YES, CHECK DATA
4726 024510 104154      ERROR    154      :CS2 INCORRECT
4727 024512 000413      BR       TST27    ;;GO ON TO NEXT TEST
4728
4729 024514 023737 003522 003462 23$:  CMP      E.DB,T.DB   :CHECK IF DATA CORRECT
4730 024522 001401      BEQ      24$      :YES, GET NEXT HEADER WORD
4731 024524 104155      ERROR    155      :DATA INCORRECT
4732 024526 005237 003624      INC      WRDCNT    :INCREMENT WORD COUNT
4733 024532 022737 000003 003624 24$:  CMP      #3,WRDCNT :CHECK IF ALL THREE WORDS CHECK
4734 024540 001327      BNE     20$      :NO, GET NEXT WORD
4735

```

\*\*\*\*\*  
 :TEST 27 READ LOOPBACK (PART 5)  
 :\*

CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
 IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06  
 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
 CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.  
 SIMULATE SECTOR PULSE, 255 ZEROES, A  
 ONE, AND A HEADER CONSISTING OF THE THREE  
 FOLLOWING WORDS.

052012  
 100520  
 052012

MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD  
 IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.

\*\*\*\*\*

```

4755 024542 000004      TST27: SCOPE
4756 024544 012737 000144 001200  MOV     #100.,$TIMES ;;DO 100. ITERATIONS
4757 024552 013702 001270      MOV     $BASE,R2    ;LOAD RK611 BASE
4758 024556 012762 100000 000000  MOV     #CCLR,RKCS1(R2) ;CLEAR RK611
4759 024564 012762 000040 000026  MOV     #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4760 024572 012762 000025 000000  MOV     #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
4761 024600 012700 000312      MOV     #50.+4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
4762 024604 012762 000440 000026 1$:  MOV     #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4763 024612 012762 000040 000026  MOV     #DMD,RKMR1(R2)
4764 024620 005300      DEC     R0
4765 024622 001370      BNE     1$
4766 024624 012762 000140 000026  MOV     #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4767 024632 012762 000040 000026  MOV     #DMD,RKMR1(R2)
4768 024640 005037 003614      CLR     PR.BIT     ;INITIALIZE PRESENT BIT AND
4769 024644 005037 003616      CLR     M1.BIT     ; PREVIOUS BIT
4770 024650 012700 000341      MOV     #225.,R0
4771 024654 004737 043540      2$:  JSR     PC,RDBIT    ;SIMULATE SYNCH
4772 024660 005300      DEC     R0
4773 024662 001374      BNE     2$
4774 024664 012737 000001 003614  MOV     #1,PR.BIT
4775 024672 004737 043540      JSR     PC,RDBIT

```

4776	024676	012701	000003			MOV	#3,R1	:LOAD NUMBER OF WORDS
4777	024702	012703	064364			MOV	#HEAD5,R3	:LOAD ADDRESS OF DATA
4778	024706	012737	000025	003500		MOV	#RDHEAD,E.CS1	:LOAD EXPECTED CS1
4779	024714	012304			5\$:	MOV	(R3)+,R4	:GET DATA
4780	024716	012700	000020			MOV	#16.,R0	:LOAD BIT COUNT
4781	024722	013737	003614	003616	6\$:	MOV	PR.BIT,M1.BIT	:STORE PREVIOUS BIT
4782	024730	006004				ROR	R4	:GET NEXT BIT
4783	024732	103403				BCS	7\$	:CHECK IF 1
4784	024734	005037	003614			CLR	PR.BIT	:NO, ZERO
4785	024740	000403				BR	8\$	:SIMULATE READ DATA
4786								
4787	024742	012737	000001	003614	7\$:	MOV	#1,PR.BIT	:ONE
4788	024750	004737	043540		8\$:	JSR	PC,RDBIT	:SIMULATE READ DATA
4789	024754	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:READ COMMAND AND STATUS REG. 1
4790	024762	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK IF CS1 CORRECT
4791	024770	001417				BEQ	9\$	:YES, SIMULATE NEXT BIT
4792	024772	012737	000003	003624		MOV	#3,WRDCNT	:LOAD WORD COUNT
4793	025000	160137	003624			SUB	R1,WRDCNT	
4794	025004	012737	000020	003622		MOV	#16.,BITCNT	:LOAD BIT COUNT
4795	025012	160037	003622			SUB	R0,BITCNT	
4796	025016	104150				ERROR	150	:CS1 INCORRECT DURING HEADER
4797	025020	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
4798	025026	000522				BR	TST30	:GO ON TO NEXT TEST
4799								
4800	025030	005300			9\$:	DEC	R0	:CHECK IF READY FOR NEXT WORD
4801	025032	001333				BNE	6\$	:NO, GET NEXT BIT
4802	025034	005301				DEC	R1	:CHECK IF HEADER FINISHED
4803	025036	001326				BNE	5\$	:NO, GET NEXT WORD
4804	025040	012700	000004			MOV	#4,R0	:LOAD COUNT FOR POSTAMBLE
4805	025044	013737	003614	003616	15\$:	MOV	PR.BIT,M1.BIT	:STORE LAST BIT
4806	025052	005037	003614			CLR	PR.BIT	:LOAD NEXT BIT
4807	025056	004737	043540			JSR	PC,RDBIT	:SIMULATE 1 BIT READ
4808	025062	005300				DEC	R0	:CHECK IF TIME FOR READY
4809	025064	001367				BNE	15\$	:NO, CONTINUE WITH POSTAMBLE
4810	025066	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:GET CURRENT CS1
4811	025074	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:GET CURRENT CS2
4812	025102	012737	000224	003500		MOV	#RDY!RDHEAD<^C<GO>>,E.CS1	:LOAD EXPECTED CS1
4813	025110	012737	000300	003510		MOV	#OR!IR,E.CS2	:LOAD EXPECTED CS2
4814	025116	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4815	025124	001401				BEQ	16\$	:YES, CHECK CS2
4816	025126	104151				ERROR	151	:CS1 INCORRECT
4817	025130	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
4818	025136	001401				BEQ	17\$	:YES, CHECK DATA
4819	025140	104152				ERROR	152	:CS2 INCORRECT
4820	025142	005037	003624		17\$:	CLR	WRDCNT	:INITIALIZE WORD COUNT
4821	025146	012703	064364			MOV	#HEAD5,R3	:GET ADDRESS OF DATA
4822	025152	012337	003522		20\$:	MOV	(R3)+,E.DB	:GET EXPECTED DATA
4823	025156	016237	000024	003462		MOV	RKDB(R2),T.DB	:GET ACTUAL DATA
4824	025164	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
4825	025172	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
4826	025200	022737	000002	003624		CMP	#2,WRDCNT	:CHECK IF LAST WORD IN DATA BUFFER
4827	025206	001003				BNE	21\$	:NO, CHECK CS1
4828	025210	012737	000100	003510		MOV	#IR,E.CS2	:STORE EXPECTED CS2
4829	025216	023737	003500	003440	21\$:	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4830	025224	001402				BEQ	22\$	:YES, CHECK CS2
4831	025226	104153				ERROR	153	:CS1 INCORRECT

```

4832 025230 000421 BR TST30 ;;GO ON TO NEXT TEST
4833
4834 025232 023737 003510 003450 22$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4835 025240 001402 BEQ 23$ ;YES, CHECK DATA
4836 025242 104154 ERROR 154 ;CS2 INCORRECT
4837 025244 000413 BR TST30 ;;GO ON TO NEXT TEST
4838
4839 025246 023737 003522 003462 23$: CMP E.DB,T.DB ;CHECK IF DATA CORRECT
4840 025254 001401 BEQ 24$ ;YES, GET NEXT HEADER WORD
4841 025256 104155 ERROR 155 ;DATA INCORRECT
4842 025260 005237 003624 24$: INC WRDCNT ;INCREMENT WORD COUNT
4843 025264 022737 000003 003624 CMP #3,WRDCNT ;CHECK IF ALL THREE WORDS CHECK
4844 025272 001327 BNE 20$ ;NO, GET NEXT WORD
4845

```

\*\*\*\*\*  
 \*TEST 30 READ HEADER IN 18 BIT MODE  
 \*\*\*\*\*

\* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
 \* IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.  
 \* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.  
 \* SIMULATE SECTOR PULSE, 255 ZEROES, A  
 \* ONE, AND A HEADER CONSISTING OF THE THREE  
 \* FOLLOWING WORDS:

177777  
 000000  
 177777

\* MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD  
 \* IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.  
 \*\*\*\*\*

```

4864 025274 000004 TST30: SCOPE
4865 025276 012737 000144 001200 MOV #100.,$TIMES ;;DO 100. ITERATIONS
4866 025304 013702 001270 MOV $BASE,R2 ;LOAD RK611
4867 025310 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4868 025316 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4869 025324 012762 010025 000000 MOV #CFMT!RDHEAD,RKCS1(R2) ;ISSUE READ HEADER (24 SECTOR FORMAT)
4870 025332 012700 000312 MOV #50.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
4871 025336 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4872 025344 012762 000040 000026 MOV #DMD,RKMR1(R2)
4873 025352 005300 DEC R0
4874 025354 001370 BNE 1$
4875 025356 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4876 025364 012762 000040 000026 MOV #DMD,RKMR1(R2)
4877 025372 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT BIT AND
4878 025376 005037 003616 CLR M1.BIT ; PREVIOUS BIT
4879 025402 012700 000377 MOV #255.,R0
4880 025406 004737 043540 2$: JSR PC,RDBIT ;SIMULATE SYNCH
4881 025412 005300 DEC R0
4882 025414 001374 BNE 2$
4883 025416 012737 000001 003614 MOV #1,PR.BIT
4884 025424 004737 043540 JSR PC,RDBIT
4885 025430 012701 000003 MOV #3,R1 ;LOAD NUMBER OF WORDS
4886 025434 012703 064326 MOV #HEAD1,R3 ;LOAD ADDRESS OF DATA
4887 025440 012737 010025 003500 MOV #CFMT!RDHEAD,E.CS1 ;LOAD EXPECTED CS1

```

4888	025446	012304			5\$:	MOV	(R3)+,R4	:GET DATA
4889	025450	012700	000020			MOV	#16.,R0	:LOAD BIT COUNT
4890	025454	013737	003614	003616	6\$:	MOV	PR.BIT,M1.BIT	:STORE PRESENT BIT
4891	025462	006004				ROR	R4	:GET NEXT BIT
4892	025464	103403				BCS	7\$	:CHECK IF 1
4893	025466	005037	003614			CLR	PR.BIT	:NO, ZERO
4894	025472	000403				BR	8\$	:SIMULATE READ DATA
4895								
4896	025474	012737	000001	003614	7\$:	MOV	#1,PR.BIT	:ONE
4897	025502	004737	043540		8\$:	JSR	PC,RDBIT	:SIMULATE READ DATA
4898	025506	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:READ COMMAND AND STATUS REG. 1
4899	025514	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK IF CS1 CORRECT
4900	025522	001417				BEQ	9\$	:YES, SIMULATE NEXT BIT
4901	025524	012737	000003	003624		MOV	#3,WRDCNT	:LOAD WORD COUNT
4902	025532	160137	003624			SUB	R1,WRDCNT	
4903	025536	012737	000020	003622		MOV	#16.,BITCNT	:LOAD BIT COUNT
4904	025544	160037	003622			SUB	R0,BITCNT	
4905	025550	104156				ERROR	156	:CS1 INCORRECT DURING HEADER
4906	025552	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611
4907	025560	000522				BR	TST31	:GO ON TO NEXT TEST
4908								
4909	025562	005300			9\$:	DEC	R0	:CHECK IF READY FOR NEXT WORD
4910	025564	001333				BNE	6\$	:NO, GET NEXT BIT
4911	025566	005301				DEC	R1	:CHECK IF HEADER FINISHED
4912	025570	001326				BNE	5\$	:NO, GET NEXT WORD
4913	025572	012700	000004			MOV	#4,R0	:LOAD COUNT FOR POSTAMBLE
4914	025576	013737	003614	003616	15\$:	MOV	PR.BIT,M1.BIT	:STORE LAST BIT
4915	025604	005037	003614			CLR	PR.BIT	:LOAD NEXT BIT
4916	025610	004737	043540			JSR	PC,RDBIT	:READ BIT
4917	025614	005300				DEC	R0	:CHECK IF TIME FOR READY
4918	025616	001367				BNE	15\$	:NO, CONTINUE WITH POSTAMBLE
4919	025620	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:GET CURRENT CS1
4920	025626	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:GET CURRENT CS2
4921	025634	012737	010224	003500		MOV	#CFMT!RDY!RDHEAD<^C<GO>>,E.CS1	:LOAD EXPECTED CS1
4922	025642	012737	000300	003510		MOV	#OR!IR,E.CS2	:LOAD EXPECTED CS2
4923	025650	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4924	025656	001401				BEQ	16\$	:YES, CHECK CS2
4925	025660	104157				ERROR	157	:CS1 INCORRECT
4926	025662	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
4927	025670	001401				BEQ	17\$	:YES, CHECK DATA
4928	025672	104160				ERROR	160	:CS2 INCORRECT
4929	025674	005037	003624		17\$:	CLR	WRDCNT	:INITIALIZE WORD COUNT
4930	025700	012703	064326			MOV	#HEAD1,R3	:GET ADDRESS OF DATA
4931	025704	012337	003522		20\$:	MOV	(R3)+,E.DB	:GET EXPECTED DATA
4932	025710	016237	000024	003462		MOV	RKDB(R2),T.DB	:GET ACTUAL DATA
4933	025716	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
4934	025724	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
4935	025732	022737	000002	003624		CMP	#2,WRDCNT	:CHECK IF LAST WORD IN DATA BUFFER
4936	025740	001003				BNE	21\$	:NO, CHECK CS1
4937	025742	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
4938	025750	023737	003500	003440	21\$:	CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
4939	025756	001402				BEQ	22\$	:YES, CHECK CS2
4940	025760	104161				ERROR	161	:CS1 INCORRECT
4941	025762	000421				BR	TST31	:GO ON TO NEXT TEST
4942								
4943	025764	023737	003510	003450	22\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT

```

4944 025772 001402          BEQ      23$          :YES, CHECK DATA BUFFER
4945 025774 104162          ERROR   162          :CS2 INCORRECT
4946 025776 000413          BR      TST31        ;;GO ON TO NEXT TEST
4947
4948 026000 023737 003522 003462 23$:  CMP      E.DB,T.DB    :CHECK DATA BUFFER CORRECT
4949 026006 001401          BEQ      24$          :YES, GET NEXT WORD
4950 026010 104163          ERROR   163          :DATA BUFFER INCORRECT
4951 026012 005237 003624          INC      WRDCNT      :INCREMENT WORD COUNT
4952 026016 022737 000003 003624 24$:  CMP      #3,WRDCNT   :CHECK IF FINISHED
4953 026024 001327          BNE     20$          :NO READ NEXT WORD
4954
4955
4956 :*****
4957 :*TEST 31          SYNCH DETECT IN READ HEADER
4958 :*
4959 :*          CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
4960 :*          IN DIAGNOSTIC MODE.  ISSUE A READ HEADER TO AN RK06
4961 :*          IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
4962 :*          CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
4963 :*          SIMULATE SECTOR PULSE AND 350 ZEROES.  MAKE
4964 :*          SURE READY REMAINS RESET AND THE SILO REMAINS
4965 :*          EMPTY.
4966 :*****
4967 026026 000004          TST31:  SCOPE
4968 026030 012737 000144 001200  MOV      #100, $TIMES  ;;DO 100. ITERATIONS
4969 026036 013702 001270          MOV      $BASE,R2     :LOAD RK611 BASE
4970 026042 012762 100000 000000  MOV      #CCLR,RKCS1(R2) :CLEAR RK611
4971 026050 012762 000040 000026  MOV      #DMD,RKMR1(R2) :PUT RK611 IN MAINT MODE
4972 026056 012762 000025 000000  MOV      #RDHEAD,RKCS1(R2) :ISSUE READ HEAD
4973 026064 012700 000312          MOV      #50.*4+2,R0   :ISSUE ENOUGH CLOCKS UNTIL READY
4974 026070 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4975 026076 012762 000040 000026  MOV      #DMD,RKMR1(R2)
4976 026104 005300          DEC      R0
4977 026106 001370          BNE     1$
4978 026110 012762 000140 000026  MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE TO SECTOR PULSE
4979 026116 012762 000040 000026  MOV      #DMD,RKMR1(R2)
4980 026124 005037 003614          CLR      PR.BIT       :INITIALIZE PRESENT AND
4981 026130 005037 003616          CLR      M1.BIT       : PREVIOUS BIT
4982 026134 012737 000025 003500  MOV      #RDHEAD,E.CS1  :LOAD EXPECTED CS1
4983 026142 012737 000100 003510  MOV      #IR,E.CS2     :LOAD EXPECTED CS2
4984 026150 005037 003622          CLR      BITCNT       :SIMULATE 350 ZEROES
4985 026154 004737 043540 2$:  JSR      PC,RDBIT
4986 026160 016237 000000 003440  MOV      RKCS1(R2),T.CS1 :STORE CS1
4987 026166 016237 000010 003450  MOV      RKCS2(R2),T.CS2 :STORE CS2
4988 026174 023737 003500 003440  CMP      E.CS1,T.CS1   :CHECK IF CS1 CORRECT
4989 026202 001402          BEQ      3$          :YES, CHECK CS2
4990 026204 104164          ERROR   164          :CS1 INCORRECT
4991 026206 000447          BR      TST32        ;;GO ON TO NEXT TEST
4992
4993 026210 023737 003510 003450 3$:  CMP      E.CS2,T.CS2   :CHECK IF CS2 CORRECT
4994 026216 001402          BEQ      4$          :YES, CHECK IF SILO EMPTY
4995 026220 104165          ERROR   165          :CS2 INCORRECT
4996 026222 000441          BR      TST32        ;;GO ON TO NEXT TEST
4997 026224 005237 003622 4$:  INC      BITCNT       :INCREMENT BIT COUNT
4998 026230 022737 000536 003622  CMP      #350.,BITCNT  :CHECK IF FINISHED
4999 026236 001346          BNE     2$          :NO, SIMULATE NEXT ZERO
    
```

```

5000 026240 005762 000024          TST    RKDB(R2)          ;READ DATA BUFFER
5001 026244 016237 000000 003440  MOV    RKCS1(R2),T.CS1  ;STORE CS1 AND CS2
5002 026252 016237 000010 003450  MOV    RKCS2(R2),T.CS2
5003 026260 012737 100224 003500  MOV    #CERR!RDY!RDHEAD<^C<GO>>,E.CS1 ;LOAD EXPECT CS1
5004 026266 012737 100100 003510  MOV    #DCK!IR,E.CS2   ;LOAD EXPECTED CS2
5005 026274 023737 003500 003440  CMP    E.CS1,T.CS1    ;CHECK FOR CONTROLLER ERROR
5006 026302 001401          BEQ    5$              ;YES, CHECK FOR DATA LATE
5007 026304 104166          ERROR  166            ;CS1 INCORRECT
5008 026306 023737 003510 003450 5$:  CMP    E.CS2,T.CS2    ;CHECK FOR DATA LATE
5009 026314 001401          BEQ    6$              ;YES, CHECK RK611
5010 026316 104167          ERROR  167            ;CS2 INCORRECT
5011 026320 012762 100000 000000 6$:  MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
    
```

```

5012
5013 *****
5014 *TEST 32          ZERO SYNCH ON READ
5015 *
5016 *          CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
5017 *          IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
5018 *          CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES,
5019 *          SIMULATE SECTOR PULSE, 255 ZEROES SHIFTED BY A HALF
5020 *          BIT TIME, A ONE, AND A HEADER CONSISTING OF THE
5021 *          THREE FOLLOWING WORDS:
5022 *
5023 *          177777
5024 *          000000
5025 *          177777
5026 *
5027 *          MAKE SURE THAT READY COMES AFTER THE THIRD WORD
5028 *          IS TRANSFERRED.  CHECK THE SILO FOR CORRECT CONTENTS.
5029 *
    
```

```

5030 *****
5031 026326 000004          TST32: SCOPE
5032 026330 012737 000144 001200  MOV    #100.,$TIMES    ;;DO 100. ITERATIONS
5033 026336 013702 001270          MOV    $BASE,R2        ;LOAD RK611 BASE
5034 026342 012762 100000 000000  MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
5035 026350 012762 000040 000026  MOV    #DMD,RKMR1(R2)  ;PUT RK611 IN DIAGNOSTIC MODE
5036 026356 012762 000025 000000  MOV    #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
5037 026364 012700 000312          MOV    #50.*4+2,R0     ;ISSUE ENOUGH CLOCKS UNTIL READY
5038 026370 012762 000440 000026 1$:  MOV    #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
5039 026376 012762 000040 000026  MOV    #DMD,RKMR1(R2)
5040 026404 005300          DEC    R0
5041 026406 001370          BNE    1$
5042 026410 012762 000140 000026  MOV    #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5043 026416 012762 000040 000026  MOV    #DMD,RKMR1(R2)
5044 026424 012762 000440 000026  MOV    #DMD!MCLK,RKMR1(R2) ;SHIFT DATA ONE HALF BIT TIME
5045 026432 012762 000040 000026  MOV    #DMD,RKMR1(R2)
5046 026440 005037 003614          CLR    PR.BIT         ;INITIALIZE PRESENT BIT AND
5047 026444 005037 003616          CLR    M1.BIT         ; PREVIOUS BIT
5048 026450 012700 000341          MOV    #225.,R0
5049 026454 004737 043540          2$:  JSR    PC,RDBIT       ;SIMULATE SYNCH
5050 026460 005300          DEC    R0
5051 026462 001374          BNE    2$
5052 026464 012737 000001 003614  MOV    #1,PR.BIT
5053 026472 004737 043540          JSR    PC,RDBIT
5054 026476 012701 000003          MOV    #3,R1          ;LOAD NUMBER OF WORDS
5055 026502 012703 064326          MOV    #HEAD1,R3      ;LOAD ADDRESS OF DATA
    
```



```

5112 027032 023737 003510 003450 22$: CMP E,CS2,T.CS2 ;CHECK CS2 CORRECT
5113 027040 001402 BEQ 23$ ;YES, CHECK DATA
5114 027042 104154 ERROR 154 ;CS2 INCORRECT
5115 027044 000413 BR TST33 ;;GO ON TO NEXT TEST
5116
5117 027046 023737 003522 003462 23$: CMP E,DB,T.DB ;CHECK IF DATA CORRECT
5118 027054 001401 BEQ 24$ ;YES, GET NEXT HEADER WORD
5119 027056 104155 ERROR 155 ;DATA INCORRECT
5120 027060 005237 003624 24$: INC WRDCNT ;INCREMENT WORD COUNT
5121 027064 022737 000003 003624 CMP #3,WRDCNT ;CHECK IF ALL THREE WORDS CHECK
5122 027072 001327 BNE 20$ ;NO, GET NEXT WORD
5123

```

.SBTTL \*\*MFM WRITE LOOPBACK TESTS

```

5124
5125
5126 *****
5127 *TEST 33 WRITE ZEROS UNTIL SECTOR PULSE WITH WRITE HEADER
5128 *
5129 * CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
5130 * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
5131 * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
5132 * CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
5133 * INDEX PULSE AND 500 DATA BITS. MAKE SURE THAT
5134 * ZEROS ARE WRITTEN. SIMULATE SECTOR PULSE AND MAKE SURE
5135 * WRITE GATE RESETS.
5136 *****
5137

```

```

5138 027074 000004 TST33: SCOPE
5139 027076 012737 000144 001200 MOV #100,$TIMES ;;DO 100. ITERATIONS
5140 027104 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5141 027110 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5142 027116 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5143 027124 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;INITIALIZE ROM
5144 027132 012762 000040 000026 MOV #DMD,RKMR1(R2)
5145 027140 012762 064630 000004 MOV #WRBUFF,RKBA(R2) ;ISSUE WRITE HEADER
5146 027146 012762 177777 000002 MOV #-1,RKWC(R2)
5147 027154 012762 000027 000000 MOV #WRHEAD,RKCS1(R2)
5148 027162 012700 002364 MOV #<256.+48.+64.+256.+10.>*2,R0 ;ISSUE ENOUGH CLOCKS
5149 ; UNTIL READY FOR INDEX PULSE
5150 027166 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
5151 027174 012762 000040 000026 MOV #DMD,RKMR1(R2)
5152 027202 005300 DEC R0
5153 027204 001370 BNE 1$
5154 027206 012700 000004 MOV #4,R0 ;SIMULATE INDEX PULSE
5155 027212 012762 000240 000026 MOV #MIND!DMD,RKMR1(R2)
5156 027220 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
5157 027226 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5158 027234 005300 DEC R0
5159 027236 001370 BNE 2$
5160 027240 012762 000040 000026 MOV #DMD,RKMR1(R2)
5161 027246 012737 062040 003524 MOV #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
5162 ; MAINT. REG
5163 027254 012700 000002 MOV #2,R0 ;WAIT FOR WRITE GATE
5164 027260 012762 000440 000026 3$: MOV #DMD!MCLK,RKMR1(R2)
5165 027266 012762 000040 000026 MOV #DMD,RKMR1(R2)
5166 027274 005300 DEC R0
5167 027276 001370 BNE 3$

```



```

5168 027300 005037 003626 CLR SECNT ;CLEAR SECTOR COUNT
5169 027304 005037 003612 CLR P1.BIT ;INITIALIZE BIT GENERATION
5170 027310 005037 003614 CLR PR.BIT
5171 027314 005037 003616 CLR M1.BIT
5172 027320 005037 003620 CLR M2.BIT
5173 027324 012700 000764 MOV #500,RO ;LOAD COUNT FOR 500 BITS
5174 027330 012737 060037 003170 MOV #EM230,EMW ;LOAD ERROR MESSAGE
5175 027336 005037 003622 CLR BITCNT ;CLEAR BIT COUNT
5176 027342 004737 042570 5$: JSR PC,WRTBIT ;WRITE ONE BIT
5177 027346 104170 ERROR 170 ;ERROR IN WRITE
5178 027350 005237 003622 INC BITCNT ;INCREMENT NUMBER OF BITS WRITTEN
5179 027354 005300 DEC RO ;CHECK IF FINISHED
5180 027356 001371 BNE 5$ ;NO, CONTINUE
5181 027360 042737 040000 003524 BIC #WRTGAT,E.MR1 ;GENERATE EXPECTED MR1
5182 027366 052737 000100 003524 BIS #MSP,E.MR1
5183 027374 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;RAISE SECTOR PULSE
5184 027402 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINT. REG. 1
5185 027410 023737 003524 003464 CMP E.MR1,T.MR1 ;STORE MAINT REG 1
5186 027416 001401 BEQ 10$ ;YES, LOWER SECTOR PULSE
5187 027420 104171 ERROR 171 ;WRITE GATE DID NOT RESET
5188 027422 042737 000100 003524 10$: BIC #MSP,E.MR1 ;GENERATE EXPECTED MR1
5189 027430 052737 040000 003524 BIS #WRTGAT,E.MR1
5190 027436 012762 000040 000026 MOV #DMD,RKMR1(R2) ;RESET SECTOR PULSE
5191 027444 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINT REG 1
5192 027452 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT
5193 027460 001401 BEQ TST34 ;:YES, GO ON TO NEXT TEST
5194 027462 104172 ERROR 172 ;WRITE GATE DID NOT SET

```

5195

5196 :\*\*\*\*\*

5197 :\*TEST 34 WRITE LOOPBACK (PART 1)

5198 :\*

5199 :\* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER

5200 :\* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06

5201 :\* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.

5202 :\* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE

5203 :\* INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER

5204 :\* CONSISTING OF THE FOLLOWING DATA, AND AN INDEX PULSE:

5205 :\*

5206 :\* 177777

5207 :\* 000000

5208 :\* 177777

5209 :\*

5210 :\* MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PULSE.

5211 :\* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

5212 :\*

5213 :\*\*\*\*\*

5214 TST34: SCOPE

5215 MOV #100,\$TIMES ;:DO 100. ITERATIONS

5216 MOV \$BASE,R2 ;:LOAD RK611 BASE

5217 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611

5218 MOV #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE

5219 MOV #HEAD1,RKBA(R2) ;:ISSUE WRITE HEADER

5220 MOV #-3,RKWC(R2)

5221 MOV #WRHEAD,RKCS1(R2)

5222 MOV #50.\*4+2,RO ;:ISSUE ENOUGH CLOCKS UNTIL

5223 ;: READY FOR INDEX PULSE

```

5224 027542 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
5225 027550 012762 000040 000026 MOV #DMD,RKMR1(R2)
5226 027556 005300 DEC R0
5227 027560 001370 BNE 1$
5228 027562 012700 000004 MOV #4,R0 ;ISSUE INDEX PULSE
5229 027566 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5230 027574 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
5231 027602 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5232 027610 005300 DEC R0
5233 027612 001370 BNE 2$
5234 027614 012762 000040 000026 MOV #DMD,RKMR1(R2)
5235 027622 012700 000010 MOV #8.,R0 ;WAIT FOR WRITE GATE
5236 027626 012762 000440 000026 3$: MOV #DMD!MCLK,RKMR1(R2)
5237 027634 012762 000040 000026 MOV #DMD,RKMR1(R2)
5238 027642 005300 DEC R0
5239 027644 001370 BNE 3$
5240 027646 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5241 027654 012762 000040 000026 MOV #DMD,RKMR1(R2)
5242 027662 005037 003626 CLR SECCNT ;INITIALIZE SECTOR COUNT
5243 027666 012737 060411 003170 MOV #EM233,EMW ;LOAD ERROR MESSAGE
5244 027674 012737 062040 003524 MOV #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
5245 ; MAINT REG 1
5246 027702 005037 003612 CLR P1.BIT ;INITIALIZE BIT GENERATION
5247 027706 005037 003614 CLR PR.BIT
5248 027712 005037 003616 CLR M1.BIT
5249 027716 005037 003620 CLR M2.BIT
5250 027722 012700 000400 MOV #256.,R0 ;SIMULATE SYNCH
5251 027726 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5252 027732 004737 042570 5$: JSR PC,WRTBIT ;WRITE ONE BIT
5253 027736 104170 ERROR 170 ;DATA INCORRECT
5254 027740 005237 003622 INC BITCNT
5255 027744 005300 DEC R0 ;CHECK IF READY FOR DATA
5256 027746 001371 BNE 5$ ;NO, GENERATE NEXT BIT
5257 027750 012737 000001 003612 MOV #1,P1.BIT ;PUT IN SYNCH BIT
5258 027756 004737 042570 JSR PC,WRTBIT
5259 027762 104170 ERROR 170 ;DATA INCORRECT
5260 027764 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5261 027770 012737 060455 003170 MOV #EM234,EMW ;LOAD ERROR MESSAGE
5262 027776 012703 064326 MOV #HEAD1,R3 ;LOAD ADDRESS OF DATA
5263 030002 012700 000003 MOV #3,R0 ;LOAD NUMBER WORDS IN HEADER
5264 030006 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
5265 030010 012701 000020 MOV #16.,R1 ;LOAD BIT COUNT
5266 030014 013737 003616 003620 12$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5267 030022 013737 003614 003616 MOV PR.BIT,M1.BIT
5268 030030 013737 003612 003614 MOV P1.BIT,PR.BIT
5269 030036 006004 ROR R4 ;SHIFT IN NEXT BIT
5270 030040 103403 BCS 14$ ;CHECK IF ONE
5271 030042 005037 003612 CLR P1.BIT ;ZERO
5272 030046 000403 BR 15$ ;CLOCK IN BIT
5273
5274 030050 012737 000001 003612 14$: MOV #1,P1.BIT ;ONE
5275 030056 004737 042570 15$: JSR PC,WRTBIT ;WRITE BIT
5276 030062 104170 ERROR 170 ;BIT INCORRECT
5277 030064 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
5278 030070 005301 DEC R1 ;CHECK IF WORD FINISHED
5279 030072 001350 BNE 12$ ;NO, CONTINUE WITH NEXT BIT
  
```

```

5280 030074 005300          DEC      R0          ;CHECK IF HEADER COMPLETE
5281 030076 001343          BNE      10$         ;NO, GET NEXT WORD
5282 030100 012701 000020     MOV      #16.,R1     ;LOAD BIT COUNT FOR NEXT WORD
5283 030104 013737 003616 003620 18$:  MOV      M1.BIT,M2.BIT ;SHIFT BITS
5284 030112 013737 003614 003616     MOV      PR.BIT,M1.BIT
5285 030120 013737 003612 003614     MOV      P1.BIT,PR.BIT
5286 030126 005037 003612          CLR      P1.BIT
5287 030132 004737 042570          JSR      PC,WRTBIT   ;WRITE ZERO
5288 030136 104170          ERROR    170        ;BIT INCORRECT
5289 030140 005237 003622          INC      BITCNT     ;INCREMENT BIT COUNT
5290 030144 005301          DEC      R1          ;CHECK IF FINISHED
5291 030146 001356          BNE      18$         ;NO, CLOCK NEXT BIT
5292 030150 012762 000240 000026     MOV      #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5293 030156 012700 000004          MOV      #4,R0
5294 030162 012762 000640 000026 20$:  MOV      #DMD!MIND!MCLK,RKMR1(R2)
5295 030170 012762 000240 000026     MOV      #DMD!MIND,RKMR1(R2)
5296 030176 005300          DEC      R0
5297 030200 001370          BNE      20$
5298 030202 012762 000040 000026     MOV      #DMD,RKMR1(R2)
5299 030210 016237 000026 003464     MOV      RKMR1(R2),T.MR1 ;GET MAINT REG 1
5300 030216 012737 022040 003524     MOV      #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
5301 030224 023737 003524 003464     CMP      E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5302 030232 001401          BEQ      25$         ;YES, CHECK IF READY SET
5303 030234 104173          ERROR    173        ;MAINT REG 1 INCORRECT
5304 030236 012700 000010 25$:  MOV      #8.,R0      ;FINISH COMMAND
5305 030242 012762 000440 000026 26$:  MOV      #DMD!MCLK,RKMR1(R2)
5306 030250 012762 000040 000026     MOV      #DMD,RKMR1(R2)
5307 030256 005300          DEC      R0
5308 030260 001370          BNE      26$
5309 030262 016237 000000 003440     MOV      RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5310 030270 012737 000226 003500     MOV      #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5311 030276 023737 003500 003440     CMP      E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5312 030304 001401          BEQ      TST35      ;;YES, GO ON TO NEXT TEST
5313 030306 104174          ERROR    174        ;CS1 INCORRECT

```

```

5314
5315 *****
5316 *TEST 35      WRITE LOOPBACK (PART 2)
5317 *
5318 *      CLEAR THE RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
5319 *      IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
5320 *      IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
5321 *      CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.  SIMULATE
5322 *      INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
5323 *      CONSISTING OF THE FOLLOWING DATA, AND AN INDEX PULSE:
5324 *
5325 *              000000
5326 *              177777
5327 *              000000
5328 *
5329 *      MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
5330 *      IS TRANSFERRED.  CHECK FOR CORRECT WRITE ENCODED DATA
5331 *      AND PRECOMPENSATION.
5332 *
5333 *****

```

```

5334 030310 000004          TST35:  SCOPE
5335 030312 012737 000144 001200     MOV      #100.,$TIMES ;:DO 100. ITERATIONS

```

5336	030320	013702	001270			MOV	\$BASE,R2	:LOAD RK611 BASE
5337	030324	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
5338	030332	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
5339	030340	012762	064334	000004		MOV	#HEAD2,RKBA(R2)	:ISSUE WRITE HEADER
5340	030346	012762	177775	000002		MOV	#-3,RKWC(R2)	
5341	030354	012762	000027	000000		MOV	#WRHEAD,RKCS1(R2)	
5342	030362	012700	000312			MOV	#50.*4+2,R0	:ISSUE ENOUGH CLOCKS UNTIL : READY FOR INDEX PULSE
5343								
5344	030366	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
5345	030374	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5346	030402	005300				DEC	R0	
5347	030404	001370				BNE	1\$	
5348	030406	012700	000004			MOV	#4,R0	:ISSUE INDEX PULSE
5349	030412	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
5350	030420	012762	000640	000026	2\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
5351	030426	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
5352	030434	005300				DEC	R0	
5353	030436	001370				BNE	2\$	
5354	030440	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5355	030446	012700	000010			MOV	#8.,R0	:WAIT FOR WRITE GATE
5356	030452	012762	000440	000026	3\$:	MOV	#DMD!MCLK,RKMR1(R2)	
5357	030460	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5358	030466	005300				DEC	R0	
5359	030470	001370				BNE	3\$	
5360	030472	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	:SIMULATE SECTOR PULSE
5361	030500	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5362	030506	005037	003626			CLR	SECCNT	:INITIALIZE SECTOR COUNT
5363	030512	012737	060411	003170		MOV	#EM233,EMW	:LOAD ERROR MESSAGE
5364	030520	012737	062040	003524		MOV	#DMD!MEWD!ECCW!WRTGAT,E.MR1	:INITIALIZE EXPECTED : MAINT REG 1
5365								
5366	030526	005037	003612			CLR	P1.BIT	:INITIALIZE BIT GENERATION
5367	030532	005037	003614			CLR	PR.BIT	
5368	030536	005037	003616			CLR	M1.BIT	
5369	030542	005037	003620			CLR	M2.BIT	
5370	030546	012700	000400			MOV	#256.,R0	:SIMULATE SYNCH
5371	030552	005037	003622			CLR	BITCNT	:INITIALIZE BIT COUNT
5372	030556	004737	042570		5\$:	JSR	PC,WRTBIT	:WRITE ONE BIT
5373	030562	104170				ERROR	170	:DATA INCORRECT
5374	030564	005237	003622			INC	BITCNT	
5375	030570	005300				DEC	R0	:CHECK IF READY FOR DATA
5376	030572	001371				BNE	5\$	:NO, GENERATE NEXT BIT
5377	030574	012737	000001	003612		MOV	#1,P1.BIT	:PUT IN SYNCH BIT
5378	030602	004737	042570			JSR	PC,WRTBIT	
5379	030606	104170				ERROR	170	:DATA INCORRECT
5380	030610	005037	003622			CLR	BITCNT	:INITIALIZE BIT COUNT
5381	030614	012737	060455	003170		MOV	#EM234,EMW	:LOAD ERROR MESSAGE
5382	030622	012703	064334			MOV	#HEAD2,R3	:LOAD ADDRESS OF DATA
5383	030626	012700	000003			MOV	#3,R0	:LOAD NUMBER WORDS IN HEADER
5384	030632	012304			10\$:	MOV	(R3)+,R4	:GET NEXT WORD
5385	030634	012701	000020			MOV	#16.,R1	:LOAD BIT COUNT
5386	030640	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
5387	030646	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5388	030654	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5389	030662	006004				ROR	R4	:SHIFT IN NEXT BIT
5390	030664	103403				BCS	14\$	:CHECK IF ONE
5391	030666	005037	003612			CLR	P1.BIT	:ZERO

```

5392 030672 000403 BR 15$ ;CLOCK IN BIT
5393
5394 030674 012737 000001 003612 14$: MOV #1,P1.BIT ;ONE
5395 030702 004737 042570 15$: JSR PC,WRTBIT ;WRITE BIT
5396 030706 104170 ERROR 170 ;BIT INCORRECT
5397 030710 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
5398 030714 005301 DEC R1 ;CHECK IF WORD FINISHED
5399 030716 001350 BNE 12$ ;NO, CONTINUE WITH NEXT BIT
5400 030720 005300 DEC R0 ;CHECK IF HEADER COMPLETE
5401 030722 001343 BNE 10$ ;NO, GET NEXT WORD
5402 030724 012701 000020 MOV #16.,R1 ;LOAD BIT COUNT FOR NEXT WORD
5403 030730 013737 003616 003620 19$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5404 030736 013737 003614 003616 MOV PR.BIT,M1.BIT
5405 030744 013737 003612 003614 MOV P1.BIT,PR.BIT
5406 030752 005037 003612 CLR P1.BIT
5407 030756 004737 042570 JSR PC,WRTBIT ;WRITE ZERO
5408 030762 104170 ERROR 170 ;BIT INCORRECT
5409 030764 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
5410 030770 005301 DEC R1 ;CHECK IF FINISHED
5411 030772 001356 BNE 18$ ;NO, CLOCK NEXT BIT
5412 030774 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5413 031002 012700 000004 MOV #4,R0
5414 031006 012762 000640 000026 20$: MOV #DMD!MIND!MCLK,RKMR1(R2)
5415 031014 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5416 031022 005300 DEC R0
5417 031024 001370 BNE 20$
5418 031026 012762 000040 000026 MOV #DMD,RKMR1(R2)
5419 031034 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
5420 031042 012737 022040 003524 MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
5421 031050 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5422 031056 001401 BEQ 25$ ;YES, CHECK IF READY SET
5423 031060 104173 ERROR 173 ;MAINT REG 1 INCORRECT
5424 031062 012700 000010 25$: MOV #8.,R0 ;FINISH COMMAND
5425 031066 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
5426 031074 012762 000040 000026 MOV #DMD,RKMR1(R2)
5427 031102 005300 DEC R0
5428 031104 001370 BNE 26$
5429 031106 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5430 031114 012737 000226 003500 MOV #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5431 031122 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5432 031130 001401 BEQ TST36 ;:YES, GO ON TO NEXT TEST
5433 031132 104174 ERROR 174 ;CS1 INCORRECT

```

```

*****
:TEST 36 WRITE LOOPBACK (PART 3)
:
: CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
: IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
: IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
: CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. STIMULATE
: INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
: CONSISTING OF THE FOLLOWING DATA, AND AN INDEX PULSE:
:
: 125252
: 052525
: 125252
:

```

5434  
5435  
5436  
5437  
5438  
5439  
5440  
5441  
5442  
5443  
5444  
5445  
5446  
5447

```

5448
5449
5450
5451
5452
5453 031134 000004
5454 031136 012737 000144 001200
5455 031144 013702 001270
5456 031150 012762 100000 000000
5457 031156 012762 000040 000026
5458 031164 012762 064342 000004
5459 031172 012762 177775 000002
5460 031200 012762 000027 000000
5461 031206 012700 000312
5462
5463 031212 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
5464 031220 012762 000040 000026 MOV #DMD,RKMR1(R2)
5465 031226 005300 DEC R0
5466 031230 001370 BNE 1$
5467 031232 012700 000004 MOV #4,R0 ;ISSUE INDEX PULSE
5468 031236 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5469 031244 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
5470 031252 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5471 031260 005300 DEC R0
5472 031262 001370 BNE 2$
5473 031264 012762 000040 000026 MOV #DMD,RKMR1(R2)
5474 031272 012700 000010 MOV #8,R0 ;WAIT FOR WRITE GATE
5475 031276 012762 000440 000026 3$: MOV #DMD!MCLK,RKMR1(R2)
5476 031304 012762 000040 000026 MOV #DMD,RKMR1(R2)
5477 031312 005300 DEC R0
5478 031314 001370 BNE 3$
5479 031316 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5480 031324 012762 000040 000026 MOV #DMD,RKMR1(R2)
5481 031332 005037 003626 CLR SECCNT ;INITIALIZE SECTOR COUNT
5482 031336 012737 060411 003170 MOV #EM233,EMW ;LOAD ERROR MESSAGE
5483 031344 012737 062040 003524 MOV #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
5484 ; MAINT REG 1
5485 031352 005037 003612 CLR P1.BIT ;INITIALIZE BIT GENERATION
5486 031356 005037 003614 CLR PR.BIT
5487 031362 005037 003616 CLR M1.BIT
5488 031366 005037 003620 CLR M2.BIT
5489 031372 012700 000400 MOV #256,R0 ;SIMULATE SYNCH
5490 031376 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5491 031402 004737 042570 5$: JSR PC,WRTBIT ;WRITE ONE BIT
5492 031406 104170 ERROR 170 ;DATA INCORRECT
5493 031410 005237 003622 INC BITCNT
5494 031414 005300 DEC R0 ;CHECK IF READY FOR DATA
5495 031416 001371 BNE 5$ ;NO, GENERATE NEXT BIT
5496 031420 012737 000001 003612 MOV #1,P1.BIT ;PUT IN SYNCH BIT
5497 031426 004737 042570 JSR PC,WRTBIT
5498 031432 104170 ERROR 170 ;DATA INCORRECT
5499 031434 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5500 031440 012737 060455 003170 MOV #EM234,EMW ;LOAD ERROR MESSAGE
5501 031446 012703 064342 MOV #HEAD3,R3 ;LOAD ADDRESS OF DATA
5502 031452 012700 000003 MOV #3,R0 ;LOAD NUMBER WORDS IN HEADER
5503 031456 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
    
```

```

5504 031460 012701 000020          MOV      #16.,R1          ;LOAD BIT COUNT
5505 031464 013737 003616 003620 12$: MOV      M1.BIT,M2.BIT    ;SHIFT BITS
5506 031472 013737 003614 003616    MOV      PR.BIT,M1.BIT
5507 031500 013737 003612 003614    MOV      P1.BIT,PR.BIT
5508 031506 006004          ROR      R4              ;SHIFT IN NEXT BIT
5509 031510 103403          BCS     14$              ;CHECK IF ONE
5510 031512 005037 003612          CLR     P1.BIT          ;ZERO
5511 031516 000403          BR      15$              ;CLOCK IN BIT
5512
5513 031520 012737 000001 003612 14$: MOV      #1,P1.BIT      ;ONE
5514 031526 004737 042570          JSR     PC,WRTBIT      ;WRITE BIT
5515 031532 104170          ERROR   170             ;BIT INCORRECT
5516 031534 005237 003622          INC     BITCNT         ;INCREMENT BIT COUNT
5517 031540 005301          DEC     R1              ;CHECK IF WORD FINISHED
5518 031542 001350          BNE     12$             ;NO, CONTINUE WITH NEXT BIT
5519 031544 005300          DEC     R0              ;CHECK IF HEADER COMPLETE
5520 031546 001343          BNE     10$             ;NO, GET NEXT WORD
5521 031550 012701 000020          MOV      #16.,R1          ;LOAD BIT COUNT FOR NEXT WORD
5522 031554 013737 003616 003620 18$: MOV      M1.BIT,M2.BIT    ;SHIFT BITS
5523 031562 013737 003614 003616    MOV      PR.BIT,M1.BIT
5524 031570 013737 003612 003614    MOV      P1.BIT,PR.BIT
5525 031576 005037 003612          CLR     P1.BIT
5526 031602 004737 042570          JSR     PC,WRTBIT      ;WRITE ZERO
5527 031606 104170          ERROR   170             ;BIT INCORRECT
5528 031610 005237 003622          INC     BITCNT         ;INCREMENT BIT COUNT
5529 031614 005301          DEC     R1              ;CHECK IF FINISHED
5530 031616 001356          BNE     18$             ;NO, CLOCK NEXT BIT
5531 031620 012762 000240 000026    MOV      #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5532 031626 012700 000004          MOV      #4,R0
5533 031632 012762 000640 000026 20$: MOV      #DMD!MIND!MCLK,RKMR1(R2)
5534 031640 012762 000240 000026    MOV      #DMD!MIND,RKMR1(R2)
5535 031646 005300          DEC     R0
5536 031650 001370          BNE     20$
5537 031652 012762 000040 000026    MOV      #DMD,RKMR1(R2)
5538 031660 016237 000026 003464    MOV      RKMR1(R2),T.MR1 ;GET MAINT REG 1
5539 031666 012737 022040 003524    MOV      #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
5540 031674 023737 003524 003464    CMP     E.MR1,T.MR1     ;CHECK MR1 CORRECT (WRITE GATE RESET)
5541 031702 001401          BEQ     25$              ;YES, CHECK IF READY SET
5542 031704 104173          ERROR   173             ;MAINT REG 1 INCORRECT
5543 031706 012700 000010          MOV      #8.,R0          ;FINISH COMMAND
5544 031712 012762 000440 000026 26$: MOV      #DMD!MCLK,RKMR1(R2)
5545 031720 012762 000040 000026    MOV      #DMD,RKMR1(R2)
5546 031726 005300          DEC     R0
5547 031730 001370          BNE     26$
5548 031732 016237 000000 003440    MOV      RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5549 031740 012737 000226 003500    MOV      #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5550 031746 023737 003500 003440    CMP     E.CS1,T.CS1     ;CHECK IF CS1 CORRECT
5551 031754 001401          BEQ     TST37            ;:YES, GO ON TO NEXT TEST
5552 031756 104174          ERROR   174             ;CS1 INCORRECT

```

```

*****
*TEST 37          WRITE LOOPBACK (PART 4)
*
*
*          CLEAR THE RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
*          IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
*          IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.

```

5553  
5554  
5555  
5556  
5557  
5558  
5559

5560  
5561  
5562  
5563  
5564  
5565  
5566  
5567  
5568  
5569  
5570  
5571  
5572 031760 000004  
5573 031762 012737 000144 001200  
5574 031770 013702 001270  
5575 031774 012762 100000 000000  
5576 032002 012762 000040 000026  
5577 032010 012762 064356 000004  
5578 032016 012762 177775 000002  
5579 032024 012762 000027 000000  
5580 032032 012700 000312  
5581  
5582 032036 012762 000440 000026 1\$:  
5583 032044 012762 000040 000026  
5584 032052 005300  
5585 032054 001370  
5586 032056 012700 000004  
5587 032062 012762 000240 000026  
5588 032070 012762 000640 000026 2\$:  
5589 032076 012762 000240 000026  
5590 032104 005300  
5591 032106 001370  
5592 032110 012762 000040 000026  
5593 032116 012700 000010  
5594 032122 012762 000440 000026 3\$:  
5595 032130 012762 000040 000026  
5596 032136 005300  
5597 032140 001370  
5598 032142 012762 000140 000026  
5599 032150 012762 000040 000026  
5600 032156 005037 003626  
5601 032162 012737 060411 003170  
5602 032170 012737 062040 003524  
5603  
5604 032176 005037 003612  
5605 032202 005037 003614  
5606 032206 005037 003616  
5607 032212 005037 003620  
5608 032216 012700 000400  
5609 032222 005037 003622  
5610 032226 004737 042570 5\$:  
5611 032232 104170  
5612 032234 005237 003622  
5613 032240 005300  
5614 032242 001371  
5615 032244 012737 000001 003612

```

**      CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
**      INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
**      CONSISTING OF THE FOLLOWING DATA, AND AND INDEX PULSE:
**
**      044444
**      022222
**      111111
**
**      MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX MODE.
**      CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.
*****
TST37: SCOPE
MOV #100.,$TIMES ;;DO 100. ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #HEAD4,RKBA(R2) ;ISSUE WRITE HEADER
MOV #-3,RKWC(R2)
MOV #WRHEAD,RKCS1(R2)
MOV #50.+4+2,RO ;ISSUE ENOUGH CLOCKS UNTIL
; READY FOR INDEX PULSE
1$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC RO
BNE 1$
MOV #4,RO ;ISSUE INDEX PULSE
2$: MOV #DMD!MIND,RKMR1(R2)
MOV #DMD!MIND!MCLK,RKMR1(R2)
MOV #DMD!MIND,RKMR1(R2)
DEC RO
BNE 2$
MOV #DMD,RKMR1(R2)
MOV #8.,RO ;WAIT FOR WRITE GATE
3$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC RO
BNE 3$
MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
MOV #DMD,RKMR1(R2)
CLR SECCNT ;INITIALIZE SECTOR COUNT
MOV #EM233,EMW ;LOAD ERROR MESSAGE
MOV #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
; MAINT REG 1
;INITIALIZE BIT GENERATION
CLR P1.BIT
CLR PR.BIT
CLR M1.BIT
CLR M2.BIT
MOV #256.,RO ;SIMULATE SYNCH
CLR BITCNT ;INITIALIZE BIT COUNT
4$: JSR PC,WRTBIT ;WRITE ONE BIT
;DATA INCORRECT
INC BITCNT
DEC RO ;CHECK IF READY FOR DATA
BNE 5$ ;NO, GENERATE NEXT BIT
MOV #1,P1.BIT ;PUT IN SYNCH BIT

```



```

5616 032252 004737 042570 JSR PC,WRTBIT
5617 032256 104170 ERROR 170 ;DATA INCORRECT
5618 032260 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5619 032264 012737 060455 003170 MOV #EM234,EMW ;LOAD ERROR MESSAGE
5620 032272 012703 064356 MOV #HEAD4,R3 ;LOAD ADDRESS OF DATA
5621 032276 012700 000003 MOV #3,R0 ;LOAD NUMBER WORDS IN HEADER
5622 032302 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
5623 032304 012701 000020 MOV #16.,R1 ;LOAD BIT COUNT
5624 032310 013737 003616 003620 12$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5625 032316 013737 003614 003616 MOV PR.BIT,M1.BIT
5626 032324 013737 003612 003614 MOV P1.BIT,PR.BIT
5627 032332 006004 ROR R4 ;SHIFT IN NEXT BIT
5628 032334 103403 BCS 14$ ;CHECK IF ONE
5629 032336 005037 003612 CLR P1.BIT ;ZERO
5630 032342 000403 BR 15$ ;CLOCK IN BIT
5631
5632 032344 012737 000001 003612 14$: MOV #1,P1.BIT ;ONE
5633 032352 004737 042570 15$: JSR PC,WRTBIT ;WRITE BIT
5634 032356 104170 ERROR 170 ;BIT INCORRECT
5635 032360 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
5636 032364 005301 DEC R1 ;CHECK IF WORD FINISHED
5637 032366 001350 BNE 12$ ;NO, CONTINUE WITH NEXT BIT
5638 032370 005300 DEC R0 ;CHECK IF HEADER COMPLETE
5639 032372 001343 BNE 10$ ;NO, GET NEXT WORD
5640 032374 012701 000020 MOV #16.,R1 ;LOAD BIT COUNT FOR NEXT WORD
5641 032400 013737 003616 003620 18$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5642 032406 013737 003614 003616 MOV PR.BIT,M1.BIT
5643 032414 013737 003612 003614 MOV P1.BIT,PR.BIT
5644 032422 005037 003612 CLR P1.BIT
5645 032426 004737 042570 JSR PC,WRTBIT ;WRITE ZERO
5646 032432 104170 ERROR 170 ;BIT INCORRECT
5647 032434 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
5648 032440 005301 DEC R1 ;CHECK IF FINISHED
5649 032442 001356 BNE 18$ ;NO, CLOCK NEXT BIT
5650 032444 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5651 032452 012700 000004 MOV #4,R0
5652 032456 012762 000640 000026 20$: MOV #DMD!MIND!MCLK,RKMR1(R2)
5653 032464 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5654 032472 005300 DEC R0
5655 032474 001370 BNE 20$
5656 032476 012762 000040 000026 MOV #DMD,RKMR1(R2)
5657 032504 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
5658 032512 012737 022040 003524 MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
5659 032520 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5660 032526 001401 BEQ 25$ ;YES, CHECK IF READY SET
5661 032530 104173 ERROR 173 ;MAINT REG 1 INCORRECT
5662 032532 012700 000010 25$: MOV #8.,R0 ;FINISH COMMAND
5663 032536 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
5664 032544 012762 000040 000026 MOV #DMD,RKMR1(R2)
5665 032552 005300 DEC R0
5666 032554 001370 BNE 26$
5667 032556 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5668 032564 012737 000226 003500 MOV #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5669 032572 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5670 032600 001401 BEQ TST40 ;:YES, GO ON TO NEXT TEST
5671 032602 104174 ERROR 174 ;CS1 INCORRECT
  
```

5672  
5673  
5674  
5675  
5676  
5677  
5678  
5679  
5680  
5681  
5682  
5683  
5684  
5685  
5686  
5687  
5688  
5689  
5690  
5691  
5692  
5693  
5694  
5695  
5696  
5697  
5698  
5699  
5700  
5701  
5702  
5703  
5704  
5705  
5706  
5707  
5708  
5709  
5710  
5711  
5712  
5713  
5714  
5715  
5716  
5717  
5718  
5719  
5720  
5721  
5722  
5723  
5724  
5725  
5726  
5727

\*\*\*\*\*  
:TEST 40 WRITE LOOPBACK (PART 5)  
\*\*\*\*\*

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

052012  
100520  
052012

MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX PULSE.  
CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

\*\*\*\*\*  
TST40: SCOPE  
\*\*\*\*\*

```
MOV #100.,$TIMES ;;DO 100. ITERATIONS
MOV $BASE,R2 ;;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #HEAD5,RKBA(R2) ;ISSUE WRITE HEADER
MOV #-3,RKWC(R2)
MOV #WRHEAD,RKCS1(R2)
MOV #50.+4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL
; READY FOR INDEX PULSE
1$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV #4,R0 ;ISSUE INDEX PULSE
MOV #DMD!MIND,RKMR1(R2)
2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
MOV #DMD!MIND,RKMR1(R2)
DEC R0
BNE 2$
MOV #DMD,RKMR1(R2)
MOV #8.,R0 ;WAIT FOR WRITE GATE
3$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 3$
MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
MOV #DMD,RKMR1(R2)
CLR SECCNT ;INITIALIZE SECTOR COUNT
MOV #EM233,EMW ;LOAD ERROR MESSAGE
MOV #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
; MAINT REG 1
; INITIALIZE BIT GENERATION
CLR P1.BIT
CLR PR.BIT
CLR M1.BIT
CLR M2.BIT
MOV #256.,R0 ;SIMULATE SYNCH
```

```

5728 033046 005037 003622          CLR      BITCNT      ;INITIALIZE BIT COUNT
5729 033052 004737 042570          JSR      PC,WRTBIT   ;WRITE ONE BIT
5730 033056 104170          ERROR    170        ;DATA INCORRECT
5731 033060 005237 003622          INC      BITCNT
5732 033064 005300          DEC      R0          ;CHECK IF READY FOR DATA
5733 033066 001371          BNE     5$          ;NO, GENERATE NEXT BIT
5734 033070 012737 000001 003612  MOV     #1,P1,BIT    ;PUT IN SYNCH BIT
5735 033076 004737 042570          JSR      PC,WRTBIT
5736 033102 104170          ERROR    170        ;DATA INCORRECT
5737 033104 005037 003622          CLR      BITCNT     ;INITIALIZE BIT COUNT
5738 033110 012737 060455 003170  MOV     #EM234,EMW   ;LOAD ERROR MESSAGE
5739 033116 012703 064364          MOV     #HEAD5,R3   ;LOAD ADDRESS OF DATA
5740 033122 012700 000003          MOV     #3,R0       ;LOAD NUMBER WORDS IN HEADER
5741 033126 012304          10$:    MOV     (R3)+,R4    ;GET NEXT WORD
5742 033130 012701 000020          MOV     #16,,R1     ;LOAD BIT COUNT
5743 033134 013737 003616 003620  12$:    MOV     M1.BIT,M2.BIT ;SHIFT BITS
5744 033142 013737 003614 003616  MOV     PR.BIT,M1.BIT
5745 033150 013737 003612 003614  MOV     P1.BIT,PR.BIT
5746 033156 006004          ROR     R4          ;SHIFT IN NEXT BIT
5747 033160 103403          BCS     14$        ;CHECK IF ONE
5748 033162 005037 003612          CLR     P1.BIT     ;ZERO
5749 033166 000403          BR      15$        ;CLOCK IN BIT
5750
5751 033170 012737 000001 003612  14$:    MOV     #1,P1.BIT   ;ONE
5752 033176 004737 042570          15$:    JSR     PC,WRTBIT   ;WRITE BIT
5753 033202 104170          ERROR    170        ;BIT INCORRECT
5754 033204 005237 003622          INC     BITCNT     ;INCREMENT BIT COUNT
5755 033210 005301          DEC     R1          ;CHECK IF WORD FINISHED
5756 033212 001350          BNE     12$        ;NO, CONTINUE WITH NEXT BIT
5757 033214 005300          DEC     R0          ;CHECK IF HEADER COMPLETE
5758 033216 001343          BNE     10$        ;NO, GET NEXT WORD
5759 033220 012701 000020          MOV     #16,,R1     ;LOAD BIT COUNT FOR NEXT WORD
5760 033224 013737 003616 003620  18$:    MOV     M1.BIT,M2.BIT ;SHIFT BITS
5761 033232 013737 003614 003616  MOV     PR.BIT,M1.BIT
5762 033240 013737 003612 003614  MOV     P1.BIT,PR.BIT
5763 033246 005037 003612          CLR     P1.BIT
5764 033252 004737 042570          JSR     PC,WRTBIT   ;WRITE ZERO
5765 033256 104170          ERROR    170        ;BIT INCORRECT
5766 033260 005237 003622          INC     BITCNT     ;INCREMENT BIT COUNT
5767 033264 005301          DEC     R1          ;CHECK IF FINISHED
5768 033266 001356          BNE     18$        ;NO, CLOCK NEXT BIT
5769 033270 012762 000240 000026  MOV     #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5770 033276 012700 000004          MOV     #4,R0
5771 033302 012762 000640 000026  20$:    MOV     #DMD!MIND!MCLK,RKMR1(R2)
5772 033310 012762 000240 000026  MOV     #DMD!MIND,RKMR1(R2)
5773 033316 005300          DEC     R0
5774 033320 001370          BNE     20$
5775 033322 012762 000040 000026  MOV     #DMD,RKMR1(R2)
5776 033330 016237 000026 003464  MOV     RKMR1(R2),T.MR1 ;GET MAINT REG 1
5777 033336 012737 022040 003524  MOV     #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
5778 033344 023737 003524 003464  CMP     E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5779 033352 001401          BEQ     25$        ;YES, CHECK IF READY SET
5780 033354 104173          ERROR    173        ;MAINT REG 1 INCORRECT
5781 033356 012700 000010          25$:    MOV     #8,,R0      ;FINISH COMMAND
5782 033362 012762 000440 000026  26$:    MOV     #DMD!MCLK,RKMR1(R2)
5783 033370 012762 000040 000026  MOV     #DMD,RKMR1(R2)
    
```

5784 033376 005300  
 5785 033400 001370  
 5786 033402 016237 000000 003440  
 5787 033410 012737 000226 003500  
 5788 033416 023737 003500 003440  
 5789 033424 001401  
 5790 033426 104174

DEC R0  
 BNE 26\$  
 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1  
 MOV #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1  
 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT  
 BEQ TST41 ;:YES, GO ON TO NEXT TEST  
 ERROR 174 ;CS1 INCORRECT

\*\*\*\*\*  
 :TEST 41 WRITE LOOPBACK (PART 6)  
 \*\*\*\*\*

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

155555  
 066666  
 155555

MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.  
 CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

5809  
 5810 033430 000004  
 5811 033432 012737 000144 001200  
 5812 033440 013702 001270  
 5813 033444 012762 100000 000000  
 5814 033452 012762 000040 000026  
 5815 033460 012762 064372 000004  
 5816 033466 012762 177775 000002  
 5817 033474 012762 000027 000000  
 5818 033502 012700 000312  
 5819  
 5820 033506 012762 000440 000026 1\$:  
 5821 033514 012762 000040 000026  
 5822 033522 005300  
 5823 033524 001370  
 5824 033526 012700 000004  
 5825 033532 012762 000240 000026  
 5826 033540 012762 000640 000026 2\$:  
 5827 033546 012762 000240 000026  
 5828 033554 005300  
 5829 033556 001370  
 5830 033560 012762 000040 000026  
 5831 033566 012700 000010  
 5832 033572 012762 000440 000026 3\$:  
 5833 033600 012762 000040 000026  
 5834 033606 005300  
 5835 033610 001370  
 5836 033612 012762 000140 000026  
 5837 033620 012762 000040 000026  
 5838 033626 005037 003626  
 5839 033632 012737 060411 003170

TST41: SCOPE  
 MOV #100.,\$TIMES ;:DO 100. ITERATIONS  
 MOV \$BASE,R2 ;LOAD RK611 BASE  
 MOV #CCLR,RKCS1(R2) ;CLEAR RK611  
 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE  
 MOV #HEAD6,RKBA(R2) ;ISSUE WRITE HEADER  
 MOV #-3,RKWC(R2)  
 MOV #WRHEAD,RKCS1(R2)  
 MOV #50.\*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL  
 ; READY FOR INDEX PULSE  
 1\$: MOV #DMD!MCLK,RKMR1(R2)  
 MOV #DMD,RKMR1(R2)  
 DEC R0  
 BNE 1\$  
 MOV #4,R0 ;ISSUE INDEX PULSE  
 MOV #DMD!MIND,RKMR1(R2)  
 2\$: MOV #DMD!MIND!MCLK,RKMR1(R2)  
 MOV #DMD!MIND,RKMR1(R2)  
 DEC R0  
 BNE 2\$  
 MOV #DMD,RKMR1(R2)  
 MOV #8.,R0 ;WAIT FOR WRITE GATE  
 3\$: MOV #DMD!MCLK,RKMR1(R2)  
 MOV #DMD,RKMR1(R2)  
 DEC R0  
 BNE 3\$  
 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE  
 MOV #DMD,RKMR1(R2)  
 CLR SECCNT ;INITIALIZE SECTOR COUNT  
 MOV #EM233,EMW ;LOAD ERROR MESSAGE

```

5840 033640 012737 062040 003524      MOV      #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
5841                                     ; MAINT REG 1
5842 033646 005037 003612      CLR      P1.BIT                       ;INITIALIZE BIT GENERATION
5843 033652 005037 003614      CLR      PR.BIT
5844 033656 005037 003616      CLR      M1.BIT
5845 033662 005037 003620      CLR      M2.BIT
5846 033666 012700 000400      MOV      #256.,R0                      ;SIMULATE SYNCH
5847 033672 005037 003622      CLR      BITCNT                       ;INITIALIZE BIT COUNT
5848 033676 004737 042570      JSR      PC,WRTBIT                     ;WRITE ONE BIT
5849 033702 104170 000000      ERROR   170                           ;DATA INCORRECT
5850 033704 005237 003622      INC      BITCNT
5851 033710 005300 000000      DEC      R0                            ;CHECK IF READY FOR DATA
5852 033712 001371 000000      BNE      5$                            ;NO, GENERATE NEXT BIT
5853 033714 012737 000001 003612      MOV      #1,P1.BIT                     ;PUT IN SYNCH BIT
5854 033722 004737 042570      JSR      PC,WRTBIT
5855 033726 104170 000000      ERROR   170                           ;DATA INCORRECT
5856 033730 005037 003622      CLR      BITCNT                       ;INITIALIZE BIT COUNT
5857 033734 012737 060455 003170      MOV      #EM234,EMW                    ;LOAD ERROR MESSAGE
5858 033742 012703 064372      MOV      #HEAD6,R3                     ;LOAD ADDRESS OF DATA
5859 033746 012700 000003      MOV      #3,R0                          ;LOAD NUMBER WORDS IN HEADER
5860 033752 012304 000000      MOV      (R3)+,R4                       ;GET NEXT WORD
5861 033754 012701 000020      MOV      #16.,R1                       ;LOAD BIT COUNT
5862 033760 013737 003616 003620 12$:  MOV      M1.BIT,M2.BIT                 ;SHIFT BITS
5863 033766 013737 003614 003616      MOV      PR.BIT,M1.BIT
5864 033774 013737 003612 003614      MOV      P1.BIT,PR.BIT
5865 034002 006004 000000      ROR      R4                            ;SHIFT IN NEXT BIT
5866 034004 103403 000000      BCS      14$                            ;CHECK IF ONE
5867 034006 005037 003612      CLR      P1.BIT                        ;ZERO
5868 034012 000403 000000      BR       15$                            ;CLOCK IN BIT
5869
5870 034014 012737 000001 003612 14$:  MOV      #1,P1.BIT                     ;ONE
5871 034022 004737 042570 15$:  JSR      PC,WRTBIT                     ;WRITE BIT
5872 034026 104170 000000      ERROR   170                           ;BIT INCORRECT
5873 034030 005237 003622      INC      BITCNT                       ;INCREMENT BIT COUNT
5874 034034 005301 000000      DEC      R1                            ;CHECK IF WORD FINISHED
5875 034036 001350 000000      BNE      12$                            ;NO, CONTINUE WITH NEXT BIT
5876 034040 005300 000000      DEC      R0                            ;CHECK IF HEADER COMPLETE
5877 034042 001343 000000      BNE      10$                            ;NO, GET NEXT WORD
5878 034044 012701 000020      MOV      #16.,R1                       ;LOAD BIT COUNT FOR NEXT WORD
5879 034050 013737 003616 003620 18$:  MOV      M1.BIT,M2.BIT                 ;SHIFT BITS
5880 034056 013737 003614 003616      MOV      PR.BIT,M1.BIT
5881 034064 013737 003612 003614      MOV      P1.BIT,PR.BIT
5882 034072 005037 003612      CLR      P1.BIT
5883 034076 004737 042570      JSR      PC,WRTBIT                     ;WRITE ZERO
5884 034102 104170 000000      ERROR   170                           ;BIT INCORRECT
5885 034104 005237 003622      INC      BITCNT                       ;INCREMENT BIT COUNT
5886 034110 005301 000000      DEC      R1                            ;CHECK IF FINISHED
5887 034112 001356 000000      BNE      18$                            ;NO, CLOCK NEXT BIT
5888 034114 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5889 034122 012700 000004 000000      MOV      #4,R0
5890 034126 012762 000640 000026 20$:  MOV      #DMD!MIND!MCLK,RKMR1(R2)
5891 034134 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
5892 034142 005300 000000      DEC      R0
5893 034144 001370 000000      BNE      20$
5894 034146 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5895 034154 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;GET MAINT REG 1

```

```

5896 034162 012737 022040 003524 MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
5897 034170 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5898 034176 001401 BEQ 25$ ;YES, CHECK IF READY SET
5899 034200 104173 ERROR 173 ;MAINT REG 1 INCORRECT
5900 034202 012700 000010 25$: MOV #8.,RO ;FINISH COMMAND
5901 034206 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
5902 034214 012762 000040 000026 MOV #DMD,RKMR1(R2)
5903 034222 005300 DEC RO
5904 034224 001370 BNE 26$
5905 034226 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5906 034234 012737 000226 003500 MOV #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5907 034242 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5908 034250 001401 BEQ TST42 ;:YES, GO ON TO NEXT TEST
5909 034252 104174 ERROR 174 ;CS1 INCORRECT
  
```

```

5910
5911 :*****
5912 :*TEST 42 WRITE LOOPBACK (PART 7)
5913 :*
  
```

```

5914 :* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
5915 :* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
5916 :* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
5917 :* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
5918 :* INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
5919 :* CONSISTING OF THE FOLLOWING DATA, AND INDEX PULSE:
  
```

```

5920 :*
5921 :* 104210
5922 :* 104210
5923 :* 104210
5924 :*
  
```

```

5925 :* MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.
5926 :* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.
5927 :*
  
```

```

5928 :*****
  
```

```

5929 034254 000004 TST42: SCOPE
5930 034256 012737 000144 001200 MOV #100.,$TIMES ;;DO 100. ITERATIONS
5931 034264 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
5932 034270 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
5933 034276 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5934 034304 012762 064400 000004 MOV #HEAD7,RKBA(R2) ;ISSUE WRITE HEADER
5935 034312 012762 177775 000002 MOV #-3,RKWC(R2)
5936 034320 012762 000027 000000 MOV #WRHEAD,RKCS1(R2)
5937 034326 012700 000312 MOV #50.*4+2,RO ;ISSUE ENOUGH CLOCKS UNTIL
5938 : ; READY FOR INDEX PULSE
5939 034332 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
5940 034340 012762 000040 000026 MOV #DMD,RKMR1(R2)
5941 034346 005300 DEC RO
5942 034350 001370 BNE 1$
5943 034352 012700 000004 MOV #4,RO ;ISSUE INDEX PULSE
5944 034356 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5945 034364 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
5946 034372 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5947 034400 005300 DEC RO
5948 034402 001370 BNE 2$
5949 034404 012762 000040 000026 MOV #DMD,RKMR1(R2)
5950 034412 012700 000010 MOV #8.,RO ;WAIT FOR WRITE GATE
5951 034416 012762 000440 000026 3$: MOV #DMD!MCLK,RKMR1(R2)
  
```

```

5952 034424 012762 000040 000026 MOV #DMD,RKMR1(R2)
5953 034432 005300 DEC R0
5954 034434 001370 BNE 3$
5955 034436 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5956 034444 012762 000040 000026 MOV #DMD,RKMR1(R2)
5957 034452 005037 003626 CLR SECCNT ;INITIALIZE SECTOR COUNT
5958 034456 012737 060411 003170 MOV #EM233,EMW ;LOAD ERROR MESSAGE
5959 034464 012737 062040 003524 MOV #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
5960 ; MAINT REG 1
5961 034472 005037 003612 CLR P1.BIT ;INITIALIZE BIT GENERATION
5962 034476 005037 003614 CLR PR.BIT
5963 034502 005037 003616 CLR M1.BIT
5964 034506 005037 003620 CLR M2.BIT
5965 034512 012700 000400 MOV #256,R0 ;SIMULATE SYNCH
5966 034516 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5967 034522 004737 042570 5$: JSR PC,WRTBIT ;WRITE ONE BIT
5968 034526 104170 ERROR 170 ;DATA INCORRECT
5969 034530 005237 003622 INC BITCNT
5970 034534 005300 DEC R0 ;CHECK IF READY FOR DATA
5971 034536 001371 BNE 5$ ;NO, GENERATE NEXT BIT
5972 034540 012737 000001 003612 MOV #1,P1.BIT ;PUT IN SYNCH BIT
5973 034546 004737 042570 JSR PC,WRTBIT
5974 034552 104170 ERROR 170 ;DATA INCORRECT
5975 034554 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5976 034560 012737 060455 003170 MOV #EM234,EMW ;LOAD ERROR MESSAGE
5977 034566 012703 064400 MOV #HEAD7,R3 ;LOAD ADDRESS OF DATA
5978 034572 012700 000003 MOV #3,R0 ;LOAD NUMBER WORDS IN HEADER
5979 034576 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
5980 034600 012701 000020 MOV #16,R1 ;LOAD BIT COUNT
5981 034604 013737 003616 003620 12$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5982 034612 013737 003614 003616 MOV PR.BIT,M1.BIT
5983 034620 013737 003612 003614 MOV P1.BIT,PR.BIT
5984 034626 006004 ROR R4 ;SHIFT IN NEXT BIT
5985 034630 103403 BCS 14$ ;CHECK IF ONE
5986 034632 005037 003612 CLR P1.BIT ;ZERO
5987 034636 000403 BR 15$ ;CLOCK IN BIT
5988
5989 034640 012737 000001 003612 14$: MOV #1,P1.BIT ;ONE
5990 034646 004737 042570 15$: JSR PC,WRTBIT ;WRITE BIT
5991 034652 104170 ERROR 170 ;BIT INCORRECT
5992 034654 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
5993 034660 005301 DEC R1 ;CHECK IF WORD FINISHED
5994 034662 001350 BNE 12$ ;NO, CONTINUE WITH NEXT BIT
5995 034664 005300 DEC R0 ;CHECK IF HEADER COMPLETE
5996 034666 001343 BNE 10$ ;NO, GET NEXT WORD
5997 034670 012701 000020 MOV #16,R1 ;LOAD BIT COUNT FOR NEXT WORD
5998 034674 013737 003616 003620 18$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5999 034702 013737 003614 003616 MOV PR.BIT,M1.BIT
6000 034710 013737 003612 003614 MOV P1.BIT,PR.BIT
6001 034716 005037 003612 CLR P1.BIT
6002 034722 004737 042570 JSR PC,WRTBIT ;WRITE ZERO
6003 034726 104170 ERROR 170 ;BIT INCORRECT
6004 034730 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
6005 034734 005301 DEC R1 ;CHECK IF FINISHED
6006 034736 001356 BNE 18$ ;NO, CLOCK NEXT BIT
6007 034740 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
  
```

```

6008 034746 012700 000004      MOV      #4,R0
6009 034752 012762 000640 000026 20$:  MOV      #DMD!MIND!MCLK,RKMR1(R2)
6010 034760 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
6011 034766 005300      DEC      R0
6012 034770 001370      BNE     20$
6013 034772 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6014 035000 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;GET MAINT REG 1
6015 035006 012737 022040 003524      MOV      #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
6016 035014 023737 003524 003464      CMP      E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
6017 035022 001401      BEQ     25$ ;YES, CHECK IF READY SET
6018 035024 104173      ERROR   173 ;MAINT REG 1 INCORRECT
6019 035026 012700 000010 25$:  MOV      #8.,R0 ;FINISH COMMAND
6020 035032 012762 000440 000026 26$:  MOV      #DMD!MCLK,RKMR1(R2)
6021 035040 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6022 035046 005300      DEC      R0
6023 035050 001370      BNE     26$
6024 035052 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6025 035060 012737 000226 003500      MOV      #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6026 035066 023737 003500 003440      CMP      E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
6027 035074 001401      BEQ     TST43 ;:YES, GO ON TO NEXT TEST
6028 035076 104174      ERROR   174 ;CS1 INCORRECT
    
```

```

6029
6030
6031 :*****
6032 :TEST 43      WRITE TWO HEADERS
6033 :
6034 :      CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
6035 :      IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
6036 :      IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
6037 :      CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.  SIMULATE
6038 :      INDEX PULSE, SECTOR PULSE, THREE WORD HEADER
6039 :      CONSISTING OF THE FOLLOWING DATA:
6040 :
6041 :          177777
6042 :          000000
6043 :          177777
6044 :
6045 :      FOLLOW THAT BY A SECTOR PULSE AND ONE THREE WORD
6046 :      HEADER CONSISTING OF THE FOLLOWING DATA:
6047 :
6048 :          000000
6049 :          177777
6050 :          000000
6051 :
6052 :      SIMULATE AN INDEX PULSE AND MAKE SURE READY COMES UP.
6053 :      CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMENSATION.
    
```

```

6054 :*****
6055 TST43:  SCOPE
6056      MOV      #100.,STIMES ;:DO 100. ITERATIONS
6057      MOV      $BASE,R2 ;:LOAD RK611 BASE
6058      MOV      #CCLR,RKCS1(R2) ;:CLEAR RK611
6059      MOV      #DMD,RKMR1(R2) ;:PUT RK611 TO DIAGNOSTIC MODE
6060      MOV      #HEAD1,RKBA(R2) ;:ISSUE WRITE HEADER
6061      MOV      #HEAD1,R3
6062      MOV      #-6,RKWC(R2)
6063      MOV      #WRHEAD,RKCS1(R2)
    
```



```

6064 035156 012700 000312      MOV      #50.*4+2,R0      ;ISSUE ENOUGH CLOCKS UNTIL
6065                                ;   READY FOR INDEX PULSE
6066 035162 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2)
6067 035170 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6068 035176 005300      DEC      R0
6069 035200 001370      BNE     1$
6070 035202 012700 000004      MOV      #4,R0           ;ISSUE INDEX PULSE
6071 035206 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
6072 035214 012762 000640 000026 2$:  MOV      #DMD!MIND!MCLK,RKMR1(R2)
6073 035222 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
6074 035230 005300      DEC      R0
6075 035232 001370      BNE     2$
6076 035234 005037 003626      CLR      SECCNT          ;CLEAR SECTOR COUNT
6077 035240 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6078 035246 012705 000002      MOV      #2,R5           ;LOAD NUMBER OF HEADERS
6079 035252 012700 000010      MOV      #8.,R0          ;WAIT FOR WRITE GATE
6080 035256 012762 000440 000026 3$:  MOV      #DMD!MCLK,RKMR1(R2)
6081 035264 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6082 035272 005300      DEC      R0
6083 035274 001370      BNE     3$
6084 035276 012762 000140 000026 4$:  MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
6085 035304 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6086 035312 012737 060411 003170      MOV      #EM233,EMW      ;LOAD ERROR MESSAGE
6087 035320 012737 062040 003524      MOV      #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
6088                                ;   MAINT REG 1
6089 035326 005037 003612      CLR      P1.BIT
6090 035332 005037 003614      CLR      PR.BIT
6091 035336 005037 003616      CLR      M1.BIT
6092 035342 005037 003620      CLR      M2.BIT
6093 035346 012700 000400      MOV      #256.,R0        ;SIMULATE SYNCH
6094 035352 005037 003622      CLR      BITCNT          ;INITIALIZE BIT COUNT
6095 035356 004737 042570 5$:  JSR      PC,WRTBIT        ;WRITE ONE BIT
6096 035362 104170      ERROR   170             ;DATA INCORRECT
6097 035364 005237 003622      INC      BITCNT
6098 035370 005300      DEC      R0              ;CHECK IF READY FOR DATA
6099 035372 001371      BNE     5$              ;NO,GENERATE NEXT BIT
6100 035374 012737 000001 003612      MOV      #1,P1.BIT        ;PUT IN SYNCH BIT
6101 035402 004737 042570      JSR      PC,WRTBIT
6102 035406 104170      ERROR   170             ;DATA INCORRECT
6103 035410 005037 003622      CLR      BITCNT          ;INITIALIZE BIT COUNT
6104 035414 012737 060455 003170      MOV      #EM234,EMW      ;LOAD ERROR MESSAGE
6105 035422 012700 000003      MOV      #3,R0           ;LOAD NUMBER OF WORDS IN HEADER
6106 035426 012304 10$:  MOV      (R3)+,R4         ;GET NEXT WORD
6107 035430 012701 000020      MOV      #16.,R1         ;LOAD BIT COUNT
6108 035434 013737 003616 003620 12$:  MOV      M1.BIT,M2.BIT    ;SHIFT BITS
6109 035442 013737 003614 003616      MOV      PR.BIT,M1.BIT
6110 035450 013737 003612 003614      MOV      P1.BIT,PR.BIT
6111 035456 006004      ROR     R4              ;SHIFT IN NEXT BIT
6112 035460 103403      BCS     14$            ;CHECK IF ONE
6113 035462 005037 003612      CLR     P1.BIT          ;ZERO
6114 035466 000403      BR     15$            ;CLOCK IN BIT
6115
6116 035470 012737 000001 003612 14$:  MOV      #1,P1.BIT        ;ONE
6117 035476 004737 042570 15$:  JSR      PC,WRTBIT        ;WRITE BIT
6118 035502 104170      ERROR   170             ;BIT INCORRECT
6119 035504 005237 003622      INC     BITCNT          ;INCREMENT BIT COUNT

```

```
6120 035510 005301 DEC R1 ;CHECK IF WORD FINISHED
6121 035512 001350 BNE 12$ ;NO, CONTINUE
6122 035514 005300 DEC R0 ;CHECK IF HEADER COMPLETE
6123 035516 001343 BNE 10$ ;NO, GET NEXT WORD
6124 035520 012701 000020 MOV #16.,R1 ;LOAD BIT COUNT FOR NEXT WORD
6125 035524 013737 003616 003620 18$: MOV M1.BIT,M2.BIT ;SHIFT BITS
6126 035532 013737 003614 003616 MOV PR.BIT,M1.BIT
6127 035540 013737 003612 003614 MOV P1.BIT,PR.BIT
6128 035546 005037 003612 CLR P1.BIT
6129 035552 004737 042570 JSR PC,WRTBIT ;WRITE ZERO
6130 035556 104170 ERROR 170 ;BIT INCORRECT
6131 035560 005237 003622 INC BITCNT ;INCREMENT
6132 035564 005301 DEC R1 ;CHECK IF READY FOR NEXT HEADER
6133 035566 001356 BNE 18$ ;NO, CONTINUE
6134 035570 005237 003626 INC SECCNT ;INCREMENT
6135 035574 005305 DEC R5 ;CHECK IF SECOND HEADER WRITTEN
6136 035576 001237 BNE 4$ ;NO, DO SECOND HEADER
6137 035600 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX PULSE
6138 035606 012700 000004 MOV #4,R0
6139 035612 012762 000640 000026 20$: MOV #DMD!MIND!MCLK,RKMR1(R2)
6140 035620 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
6141 035626 005300 DEC R0
6142 035630 001370 BNE 20$
6143 035632 012762 000040 000026 MOV #DMD,RKMR1(R2)
6144 035640 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
6145 035646 012737 022040 003524 MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
6146 035654 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
6147 035662 001401 BEQ 25$ ;YES, CHECK IF READY SET
6148 035664 104173 ERROR 173 ;MAINT REG 1 INCORRECT
6149 035666 012700 000010 25$: MOV #8.,R0 ;FINISH COMMAND
6150 035672 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
6151 035700 012762 000040 000026 MOV #DMD,RKMR1(R2)
6152 035706 005300 DEC R0
6153 035710 001370 BNE 26$
6154 035712 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6155 035720 012737 000226 003500 MOV #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6156 035726 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
6157 035734 001401 BEQ TST44 ;:YES, GO ON TO NEXT TEST
6158 035736 104174 ERROR 174
```

```
6159
6160
6161 :*****
6162 :*TEST 44 DATA FIELD FILLING ON WRITE HEADER
6163 :*
6164 :* CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
6165 :* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
6166 :* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, AND
6167 :* SPECIFY TWO 3 WORD HEADERS CONSISTING OF THE
6168 :* FOLLOWING DATA:
6169 :*
6170 :* 125252
6171 :* 052525
6172 :* 125252
6173 :* 052525
6174 :* 125252
6175 :* 052525
6176 :*
```

```
6176      ;*      MAKE SURE THE DATA SYNCH ANY OTHER BITS OF DATA AND
6177      ;*      ECC FIELD ARE WRITTEN CORRECTLY.
6178      ;*
6179      ;*****
6180 035740 000004 TST44: SCOPE
6181 035742 012737 000144 001200 MOV #100.,$TIMES ;;DO 100. ITERATIONS
6182 035750 013702 001270 MOV $BASE,R2 ;;LOAD RK611 BASE
6183 035754 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
6184 035762 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
6185 035770 012762 064342 000004 MOV #HEAD3,RKBA(R2) ;ISSUE READ HEADER
6186 035776 012703 064342 MOV #HEAD3,R3
6187 036002 012762 177772 000002 MOV #-2*3,RKWC(R2)
6188 036010 012762 000027 000000 MOV #WRHEAD,RKCS1(R2)
6189 036016 012700 000312 MOV #50.*4+2,R0 ;ISSUE CLOCKS UNTIL READY
6190      ; FOR INDEX PULSE
6191 036022 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
6192 036030 012762 000040 000026 MOV #DMD,RKMR1(R2)
6193 036036 005300 DEC R0
6194 036040 001370 BNE 1$
6195 036042 012700 000004 MOV #4,R0 ;ISSUE INDEX PULSE
6196 036046 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
6197 036054 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
6198 036062 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
6199 036070 005300 DEC R0
6200 036072 001370 BNE 2$
6201 036074 012762 000040 000026 MOV #DMD,RKMR1(R2)
6202 036102 012700 000010 MOV #8.,R0 ;WAIT FOR WRITE GATE
6203 036106 012762 000440 000026 3$: MOV #DMD!MCLK,RKMR1(R2)
6204 036114 012762 000040 000026 MOV #DMD,RKMR1(R2)
6205 036122 005300 DEC R0
6206 036124 001370 BNE 3$
6207 036126 005037 003626 CLR SECCNT ;CLEAR SECTOR COUNT
6208 036132 012705 000002 MOV #2,R5 ;LOAD NUMBER OF HEADERS
6209 036136 012762 000140 000026 4$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
6210 036144 012762 000040 000026 MOV #DMD,RKMR1(R2)
6211 036152 012737 060411 003170 MOV #EM233,EMW ;LOAD ERROR MESSAGE
6212 036160 012737 062040 003524 MOV #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
6213      ; MAINT REG 1
6214 036166 005037 003612 CLR P1.BIT
6215 036172 005037 003614 CLR PR.BIT
6216 036176 005037 003616 CLR M1.BIT
6217 036202 005037 003620 CLR M2.BIT
6218 036206 012700 000400 MOV #256.,R0 ;SIMULATE SYNCH
6219 036212 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
6220 036216 004737 042570 5$: JSR PC,WRTBIT ;WRITE ONE BIT
6221 036222 104170 ERROR 170 ;DATA INCORRECT
6222 036224 005237 003622 INC BITCNT
6223 036230 005300 DEC R0 ;CHECK IF READY FOR DATA
6224 036232 001371 BNE 5$ ;NO, GENERATE NEXT BIT
6225 036234 012737 000001 003612 MOV #1,P1.BIT ;PUT IN SYNCH BIT
6226 036242 004737 042570 JSR PC,WRTBIT
6227 036246 104170 ERROR 170 ;DATA INCORRECT
6228 036250 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
6229 036254 012737 060455 003170 MOV #EM234,EMW ;LOAD ERROR MESSAGE
6230 036262 012700 000003 MOV #3,R0 ;LOAD NUMBER OF WORDS IN HEADER
6231 036266 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
```

6232	036270	012701	000020			MOV	#16.,R1	:LOAD BIT COUNT
6233	036274	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
6234	036302	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6235	036310	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6236	036316	006004				ROR	R4	:SHIFT IN NEXT BIT
6237	036320	103403				BCS	14\$	:CHECK IF ONE
6238	036322	005037	003612			CLR	P1.BIT	:ZERO
6239	036326	000403				BR	15\$	:CLOCK IN BIT
6240								
6241	036330	012737	000001	003612	14\$:	MOV	#1,P1.BIT	:ONE
6242	036336	004737	042570		15\$:	JSR	PC,WRTBIT	:WRITE BIT
6243	036342	104170				ERROR	170	:BIT INCORRECT
6244	036344	005237	003622			INC	BITCNT	:INCREMENT BIT COUNT
6245	036350	005301				DEC	R1	:CHECK IF WORD FINISHED
6246	036352	001350				BNE	12\$	:NO, CONTINUE
6247	036354	005300				DEC	R0	:CHECK IF HEADER COMPLETE
6248	036356	001343				BNE	10\$	:NO, GET NEXT WORD
6249	036360	012737	060707	003170		MOV	#EM237,EMW	:LOAD ERROR MESSAGE
6250	036366	012701	000477			MOV	#64.+255.,R1	:LOAD COUNT
6251	036372	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
6252	036400	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6253	036406	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6254	036414	005037	003612			CLR	P1.BIT	
6255	036420	004737	042570			JSR	PC,WRTBIT	:WRITE ZERO
6256	036424	104170				ERROR	170	:BIT INCORRECT
6257	036426	005301				DEC	R1	:CHECK IF READY FOR DATA
6258	036430	001360				BNE	18\$	:NO, CONTINUE
6259	036432	012737	000001	003612		MOV	#1,P1.BIT	:SET SYNCH BIT
6260	036440	004737	042570			JSR	PC,WRTBIT	:WRITE SYNCH BIT
6261	036444	104170				ERROR	170	:SYNCH BIT INCORRECT
6262	036446	012737	060755	003170		MOV	#EM238,EMW	:LOAD ERROR MESSAGE
6263	036454	005037	003622			CLR	BITCNT	:CLEAR BIT COUNT
6264	036460	012701	007777			MOV	#256.+16.-1,R1	:LOAD COUNT
6265	036464	013737	003616	003620	20\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
6266	036472	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6267	036500	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6268	036506	005037	003612			CLR	P1.BIT	
6269	036512	004737	042570			JSR	PC,WRTBIT	:WRITE ZEROS
6270	036516	104170				ERROR	170	:BIT INCORRECT
6271	036520	005237	003622			INC	BITCNT	:INCREMENT BIT COUNT
6272	036524	005301				DEC	R1	:CHECK IF READY FOR ECC
6273	036526	001356				BNE	20\$	:NO, CONTINUE
6274	036530	012701	000040			MOV	#32.,R1	:LOAD COUNT
6275	036534	042737	020000	003524		BIC	#ECCW,E.MR1	:RESET ECCW BIT
6276	036542	004737	042570		21\$:	JSR	PC,WRTBIT	:WRITE ECC FIELD
6277	036546	104170				ERROR	170	:BIT INCORRECT
6278	036550	005237	003622			INC	BITCNT	:INCREMENT COUNT
6279	036554	005301				DEC	R1	:CHECK IF READY FOR POST AMBLE
6280	036556	001371				BNE	21\$	:NO CONTINUE
6281	036560	052737	020000	003524		BIS	#ECCW,E.MR1	:SET ECCW
6282	036566	012701	000012			MOV	#10.,R1	:CHECK IF READY FOR NEXT HEADER
6283	036572	004737	042570		22\$:	JSR	PC,WRTBIT	:WRITE POSTAMBLE
6284	036576	104170				ERROR	170	:BIT INCORRECT
6285	036600	005237	003622			INC	BITCNT	:INCREMENT COUNT
6286	036604	005301				DEC	R1	:CHECK IF READY FOR NEXT HEADER
6287	036606	001371				BNE	22\$	:NO, COMPLETE POSTAMBLE

```

6288 036610 005237 003626 INC SECCNT ;INCREMENT COUNT
6289 036614 005305 DEC R5 ;CHECK IF ALL HEADERS RECEIVED
6290 036616 001402 BEQ 23$ ;YES, SIMULATE INDEX
6291 036620 000137 036136 JMP 4$ ;NO GET NEXT HEADER
6292
6293 036624 012762 000240 000026 23$: MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX PULSE
6294 036632 012700 000004 MOV #4,RO
6295 036636 012762 000640 000026 25$: MOV #DMD!MIND!MCLK,RKMR1(R2)
6296 036644 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
6297 036652 005300 DEC RO
6298 036654 001370 BNE 25$
6299 036656 012762 000040 000026 MOV #DMD,RKMR1(R2)
6300 036664 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
6301 036672 012737 022040 003524 MOV #MEWD!ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
6302 036700 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK IF MR1 CORRECT
6303 ; (WRITE GATE RESET)
6304 036706 001401 BEQ 28$ ;YES, CHECK IF READY SET
6305 036710 104173 ERROR 173 ;MAINTENANCE REG 1 INCORRECT
6306 036712 012700 000010 28$: MOV #8.,RO ;FINISH COMMAND
6307 036716 012762 000440 000026 29$: MOV #DMD!MCLK,RKMR1(R2)
6308 036724 012762 000040 000026 MOV #DMD,RKMR1(R2)
6309 036732 005300 DEC RO
6310 036734 001370 BNE 29$
6311 036736 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6312 036744 012737 000226 003500 MOV #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6313 036752 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
6314 036760 001401 BEQ TST45 ;:YES, GO TO NEXT TEST
6315 036762 104174 ERROR 174 ;CS1 INCORRECT
    
```

```

*****
: *TEST 45 WRITE HEADER FOR 26 SECTORS
: *
: * CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
: * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
: * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0, SPECIFYING
: * 66 WORDS. MAKE SURE ALL 26 SECTORS ARE WRITTEN CORRECTLY.
: *
*****
    
```

```

6326 036764 000004 TST45: SCOPE
6327 036766 012737 000010 001200 MOV #10,$TIMES ;:DO 10 ITERATIONS
6328 036774 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
6329 037000 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
6330 037006 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
6331 037014 012762 064406 000004 MOV #NPRBUF,RKBA(R2) ;ISSUE READ HEADER
6332 037022 012703 064406 MOV #NPRBUF,R3
6333 037026 012762 177676 000002 MOV #-22.*3,RKWC(R2)
6334 037034 012762 000027 000000 MOV #WRHEAD,RKCS1(R2)
6335 037042 012700 000312 MOV #50.*4+2,RO ;ISSUE CLOCKS UNTIL READY
6336 ; FOR INDEX PULSE
6337 037046 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
6338 037054 012762 000040 000026 MOV #DMD,RKMR1(R2)
6339 037062 005300 DEC RO
6340 037064 001370 BNE 1$
6341 037066 012700 000004 MOV #4,RO ;ISSUE INDEX PULSE
6342 037072 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
6343 037100 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
    
```

```

6344 037106 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
6345 037114 005300                      DEC      R0
6346 037116 001370                      BNE     2$
6347 037120 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6348 037126 012700 000010                      MOV      #8.,R0          ;WAIT FOR WRITE GATE
6349 037132 012762 000440 000026 3$:   MOV      #DMD!MCLK,RKMR1(R2)
6350 037140 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6351 037146 005300                      DEC      R0
6352 037150 001370                      BNE     3$
6353 037152 005037 003626                      CLR      SECCNT          ;CLEAR SECTOR COUNT
6354 037156 012705 000026                      MOV      #22.,R5        ;LOAD NUMBER OF HEADERS
6355 037162 012762 000140 000026 4$:   MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
6356 037170 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6357 037176 012737 060411 003170      MOV      #EM233,EMW     ;LOAD ERROR MESSAGE
6358 037204 012737 062040 003524      MOV      #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
6359                                     ; MAINT REG 1
6360 037212 005037 003612                      CLR      P1.BIT
6361 037216 005037 003614                      CLR      PR.BIT
6362 037222 005037 003616                      CLR      M1.BIT
6363 037226 005037 003620                      CLR      M2.BIT
6364 037232 012700 000400                      MOV      #256.,R0       ;SIMULATE SYNCH
6365 037236 005037 003622                      CLR      BITCNT        ;INITIALIZE BIT COUNT
6366 037242 004737 042570                      JSR      PC,WRTBIT     ;WRITE ONE BIT
6367 037246 104170                      ERROR    170          ;DATA INCORRECT
6368 037250 005237 003622                      INC      BITCNT
6369 037254 005300                      DEC      R0            ;CHECK IF READY FOR DATA
6370 037256 001371                      BNE     5$            ;NO, GENERATE NEXT BIT
6371 037260 012737 000001 003612      MOV      #1,P1.BIT    ;PUT IN SYNCH BIT
6372 037266 004737 042570                      JSR      PC,WRTBIT
6373 037272 104170                      ERROR    170          ;DATA INCORRECT
6374 037274 005037 003622                      CLR      BITCNT        ;INITIALIZE BIT COUNT
6375 037300 012737 060455 003170      MOV      #EM234,EMW   ;LOAD ERROR MESSAGE
6376 037306 012700 000003                      MOV      #3,R0        ;LOAD NUMBER OF WORDS IN HEADER
6377 037312 012304                      MOV      (R3)+,R4     ;GET NEXT WORD
6378 037314 012701 000020                      MOV      #16.,R1     ;LOAD BIT COUNT
6379 037320 013737 003616 003620 12$:  MOV      M1.BIT,M2.BIT ;SHIFT BITS
6380 037326 013737 003614 003616      MOV      PR.BIT,M1.BIT
6381 037334 013737 003612 003614      MOV      P1.BIT,PR.BIT
6382 037342 006004                      ROR      R4            ;SHIFT IN NEXT BIT
6383 037344 103403                      BCS     14$          ;CHECK IF ONE
6384 037346 005037 003612                      CLR      P1.BIT      ;ZERO
6385 037352 000403                      BR      15$          ;CLOCK IN BIT
6386
6387 037354 012737 000001 003612 14$:  MOV      #1,P1.BIT    ;ONE
6388 037362 004737 042570 15$:  JSR      PC,WRTBIT    ;WRITE BIT
6389 037366 104170                      ERROR    170          ;BIT INCORRECT
6390 037370 005237 003622                      INC      BITCNT        ;INCREMENT BIT COUNT
6391 037374 005301                      DEC      R1            ;CHECK IF WORD FINISHED
6392 037376 001350                      BNE     12$          ;NO, CONTINUE
6393 037400 005300                      DEC      R0            ;CHECK IF HEADER COMPLETE
6394 037402 001343                      BNE     10$          ;NO, GET NEXT WORD
6395 037404 012737 060707 003170      MOV      #EM237,EMW   ;LOAD ERROR MESSAGE
6396 037412 012701 000477                      MOV      #64.+255.,R1 ;LOAD COUNT
6397 037416 013737 003616 003620 18$:  MOV      M1.BIT,M2.BIT ;SHIFT BITS
6398 037424 013737 003614 003616      MOV      PR.BIT,M1.BIT
6399 037432 013737 003612 003614      MOV      P1.BIT,PR.BIT
  
```

6400	037440	005037	003612			CLR	P1.BIT	
6401	037444	004737	042570			JSR	PC,WRTBIT	:WRITE ZERO
6402	037450	104170				ERROR	170	:BIT INCORRECT
6403	037452	005301				DEC	R1	:CHECK IF READY FOR DATA
6404	037454	001360				BNE	18\$	:NO, CONTINUE
6405	037456	012737	000001	003612		MOV	#1,P1.BIT	:SET SYNCH BIT
6406	037464	004737	042570			JSR	PC,WRTBIT	:WRITE SYNCH BIT
6407	037470	104170				ERROR	170	:SYNCH BIT INCORRECT
6408	037472	012737	060755	003170		MOV	#EM238,EMW	:LOAD ERROR MESSAGE
6409	037500	005037	003622			CLR	BITCNT	:CLEAR BIT COUNT
6410	037504	012701	007777			MOV	#256.*16.-1,R1	:LOAD COUNT
6411	037510	013737	003616	003620	27\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
6412	037516	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6413	037524	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6414	037532	005037	003612			CLR	P1.BIT	
6415	037536	004737	042570			JSR	PC,WRTBIT	:WRITE ZEROS
6416	037542	104170				ERROR	170	:BIT INCORRECT
6417	037544	005237	003622			INC	BITCNT	:INCREMENT BIT COUNT
6418	037550	005301				DEC	R1	:CHECK IF READY FOR ECC
6419	037552	001356				BNE	20\$	:NO, CONTINUE
6420	037554	012701	000040			MOV	#32.,R1	:LOAD COUNT
6421	037560	042737	020000	003524		BIC	#ECCW,E.MR1	:RESET ECCW BIT
6422	037566	004737	042570		21\$:	JSR	PC,WRTBIT	:WRITE ECC FIELD
6423	037572	104170				ERROR	170	:BIT INCORRECT
6424	037574	005237	003622			INC	BITCNT	:INCREMENT COUNT
6425	037600	005301				DEC	R1	:CHECK IF READY FOR POST AMBLE
6426	037602	001371				BNE	21\$	:NO CONTINUE
6427	037604	052737	020000	003524		BIS	#ECCW,E.MR1	:SET ECCW
6428	037612	012701	000012			MOV	#10.,R1	:CHECK IF READY FOR NEXT HEADER
6429	037616	004737	042570		22\$:	JSR	PC,WRTBIT	:WRITE POSTAMBLE
6430	037622	104170				ERROR	170	:BIT INCORRECT
6431	037624	005237	003622			INC	BITCNT	:INCREMENT COUNT
6432	037630	005301				DEC	R1	:CHECK IF READY FOR NEXT HEADER
6433	037632	001371				BNE	22\$	:NO, COMPLETE POSTAMBLE
6434	037634	005237	003626			INC	SECCNT	:INCREMENT COUNT
6435	037640	005305				DEC	R5	:CHECK IF ALL HEADERS RECEIVED
6436	037642	001402				BEQ	23\$	:YES, SIMULATE INDEX
6437	037644	000137	037162			JMP	4\$	:NO GET NEXT HEADER
6438								
6439	037650	012762	000240	000026	23\$:	MOV	#DMD!MIND,RKMR1(R2)	:SIMULATE INDEX PULSE
6440	037656	012700	000004			MOV	#4,R0	
6441	037662	012762	000640	000026	25\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
6442	037670	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
6443	037676	005300				DEC	R0	
6444	037700	001370				BNE	25\$	
6445	037702	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6446	037710	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:GET MAINT REG 1
6447	037716	012737	022040	003524		MOV	#MEWD!ECCW!DMD,E.MR1	:LOAD EXPECTED MR1
6448	037724	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK IF MR1 CORRECT
6449								: (WRITE GATE RESET)
6450	037732	001401				BEQ	28\$	:YES, CHECK IF READY SET
6451	037734	104173				ERROR	173	:MAINTENANCE REG 1 INCORRECT
6452	037736	012700	000010		28\$:	MOV	#8.,R0	:FINISH COMMAND
6453	037742	012762	000440	000026	29\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6454	037750	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6455	037756	005300				DEC	R0	

6456	037760	001370			BNE	29\$	
6457	037762	016237	000000	003440	MOV	RKCS1(R2),T.CS1	;GET COMMAND AND STATUS REG 1
6458	037770	012737	000226	003500	MOV	#RDY!WRHEAD<^C<GO>>,E.CS1	;LOAD EXPECTED CS1
6459	037776	023737	003500	003440	CMP	E.CS1,T.CS1	;CHECK IF CS1 CORRECT
6460	040004	001401			BEQ	TST46	::YES, GO TO NEXT TEST
6461	040006	104174			ERROR	174	;CS1 INCORRECT

\*\*\*\*\*  
 \*TEST 46 WRITE HEADER IN 24 SECTOR FORMAT  
 \*\*\*\*\*

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER OF SIX WORDS IN 24 SECTOR FORMAT TO AN RK06, CYLINDER 0, HEAD 0, DRIVE 0. CLOCK THROUGH THE SEEK AND DRIVE CLEAR MESSAGES, SIMULATE INDEX PULSE, SECTOR PULSE, 3 HEADER WORDS, SYNCH AND DATA, ANOTHER SECTOR PULSE, 3 HEADER WORDS, AND AN INDEX PULSE. CHECK DATA WRITTEN TO MAKE SURE ONLY LOW 16 BITS OF SILO ARE USED.

\*\*\*\*\*

6476	040010	000004			TST46:	SCOPE	
6477	040012	012737	000144	001200	MOV	#100.,\$TIMES	::DO 100. ITERATIONS
6478	040020	013702	001270		MOV	\$BASE,R2	;LOAD RK611 BASE
6479	040024	012762	100000	000000	MOV	#CCLR,RKCS1(R2)	;CLEAR RK611
6480	040032	012762	000040	000026	MOV	#DMD,RKMR1(R2)	;PUT RK611 TO DIAGNOSTIC MODE
6481	040040	012762	064326	000004	MOV	#HEAD1,RKBA(R2)	;ISSUE WRITE HEADER
6482	040046	012703	064326		MOV	#HEAD1,R3	
6483	040052	012762	177772	000002	MOV	#-6,RKWC(R2)	
6484	040060	012762	010027	000000	MOV	#CFMT!WRHEAD,RKCS1(R2)	
6485	040066	012700	000312		MOV	#50.*4+2,R0	;ISSUE ENOUGH CLOCKS UNTIL ; READY FOR INDEX PULSE
6486							
6487	040072	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)
6488	040100	012762	000040	000026		MOV	#DMD,RKMR1(R2)
6489	040106	005300				DEC	R0
6490	040110	001370				BNE	1\$
6491	040112	012700	000004			MOV	#4,R0 ;ISSUE INDEX PULSE
6492	040116	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)
6493	040124	012762	000640	000026	2\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)
6494	040132	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)
6495	040140	005300				DEC	R0
6496	040142	001370				BNE	2\$
6497	040144	005037	003626			CLR	SECCNT ;CLEAR SECTOR COUNT
6498	040150	012762	000040	000026		MOV	#DMD,RKMR1(R2)
6499	040156	012705	000002			MOV	#2,R5 ;LOAD NUMBER OF HEADERS
6500	040162	012700	000010			MOV	#8.,R0 ;WAIT FOR WRITE GATE
6501	040166	012762	000440	000026	3\$:	MOV	#DMD!MCLK,RKMR1(R2)
6502	040174	012762	000040	000026		MOV	#DMD,RKMR1(R2)
6503	040202	005300				DEC	R0
6504	040204	001370				BNE	3\$
6505	040206	012762	000140	000026	4\$:	MOV	#DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
6506	040214	012762	000040	000026		MOV	#DMD,RKMR1(R2)
6507	040222	012737	061014	003170		MOV	#EM239,EMW ;LOAD ERROR MESSAGE
6508	040230	012737	062040	003524		MOV	#DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED ; MAINT REG 1
6509							
6510	040236	005037	003612			CLR	P1.BIT
6511	040242	005037	003614			CLR	PR.BIT



6512	040246	005037	003616			CLR	M1.BIT	
6513	040252	005037	003620			CLR	M2.BIT	
6514	040256	012700	000400			MOV	#256.,R0	;SIMULATE SYNCH
6515	040262	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
6516	040266	004737	042570		5\$:	JSR	PC,WRTBIT	;WRITE ONE BIT
6517	040272	104170				ERROR	170	;DATA INCORRECT
6518	040274	005237	003622			INC	BITCNT	
6519	040300	005300				DEC	R0	;CHECK IF READY FOR DATA
6520	040302	001371				BNE	5\$	;NO,GENERATE NEXT BIT
6521	040304	012737	000001	003612		MOV	#1,P1.BIT	;PUT IN SYNCH BIT
6522	040312	004737	042570			JSR	PC,WRTBIT	
6523	040316	104170				ERROR	170	;DATA INCORRECT
6524	040320	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
6525	040324	012737	061106	003170		MOV	#EM240,EMW	;LOAD ERROR MESSAGE
6526	040332	012700	000003			MOV	#3,R0	;LOAD NUMBER OF WORDS IN HEADER
6527	040336	012304			10\$:	MOV	(R3)+,R4	;GET NEXT WORD
6528	040340	012701	000020			MOV	#16.,R1	;LOAD BIT COUNT
6529	040344	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
6530	040352	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6531	040360	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6532	040366	006004				ROR	R4	;SHIFT IN NEXT BIT
6533	040370	103403				BCS	14\$	;CHECK IF ONE
6534	040372	005037	003612			CLR	P1.BIT	;ZERO
6535	040376	000403				BR	15\$	;CLOCK IN BIT
6536								
6537	040400	012737	000001	003612	14\$:	MOV	#1,P1.BIT	;ONE
6538	040406	004737	042570		15\$:	JSR	PC,WRTBIT	;WRITE BIT
6539	040412	104170				ERROR	170	;BIT INCORRECT
6540	040414	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT
6541	040420	005301				DEC	R1	;CHECK IF WORD FINISHED
6542	040422	001350				BNE	12\$	;NO, CONTINUE
6543	040424	005300				DEC	R0	;CHECK IF HEADER COMPLETE
6544	040426	001343				BNE	10\$	;NO, GET NEXT WORD
6545	040430	012701	000020			MOV	#16.,R1	;LOAD BIT COUNT FOR NEXT WORD
6546	040434	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
6547	040442	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6548	040450	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6549	040456	005037	003612			CLR	P1.BIT	
6550	040462	004737	042570			JSR	PC,WRTBIT	;WRITE ZERO
6551	040466	104170				ERROR	170	;BIT INCORRECT
6552	040470	005237	003622			INC	BITCNT	;INCREMENT
6553	040474	005301				DEC	R1	;CHECK IF READY FOR NEXT HEADER
6554	040476	001356				BNE	18\$	;NO, CONTINUE
6555	040500	005237	003626			INC	SECCNT	;INCREMENT
6556	040504	005305				DEC	R5	;CHECK IF SECOND HEADER WRITTEN
6557	040506	001237				BNE	4\$	;NO, DO SECOND HEADER
6558	040510	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	;SIMULATE INDEX PULSE
6559	040516	012700	000004			MOV	#4,R0	
6560	040522	012762	000640	000026	20\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
6561	040530	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
6562	040536	005300				DEC	R0	
6563	040540	001370				BNE	20\$	
6564	040542	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6565	040550	016237	000026	003464		MOV	RKMR1(R2),T.MR1	;GET MAINT REG 1
6566	040556	012737	022040	003524		MOV	#MEWD!ECCW!DMD,E.MR1	;LOAD EXPECTED MR1
6567	040564	023737	003524	003464		CMP	E.MR1,T.MR1	;CHECK MR1 CORRECT (WRITE GATE RESET)

```

6568 040572 001401 BEQ 25$ ;YES, CHECK IF READY SET
6569 040574 104173 ERROR 173 ;MAINT REG 1 INCORRECT
6570 040576 012700 000010 25$: MOV #8.,R0 ;FINISH COMMAND
6571 040602 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
6572 040610 012762 000040 000026 MOV #DMD,RKMR1(R2)
6573 040616 005300 DEC R0
6574 040620 001370 BNE 26$
6575 040622 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6576 040630 012737 010226 003500 MOV #RDY!CFMT!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6577 040636 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
6578 040644 001401 BEQ TST47 ;:YES, GO ON TO NEXT TEST
6579 040646 104174 ERROR 174
    
```

.SBTTL \*\*TYPE B INSTRUCTION ERRORS

```

*****
:TEST 47 FORMAT ERROR (PART 1)
:
: CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
: CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN
: RK06, IN 26 SECTOR FORMAT, CYLINDER 43, HEAD 0, DRIVE 0.
: CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN OFF
: MAINTENANCE MODE. MAKE SURE FORMAT ERROR,
: DRIVE AVAILABLE AND CONTROLLER ERROR SET.
:
*****
    
```

```

6593 *****
6594 040650 000004 TST47: SCOPE
6595 040652 012737 000144 001200 MOV #100.,$TIMES ;:DO 100. ITERATIONS
6596 040660 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
6597 040664 012762 000040 000010 MOV #SCLR,RKCS2(R2) ;CLEAR RK611 SUBSYSTEM
6598 040672 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
6599 040700 012762 000043 000020 MOV #43,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG
6600 040706 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
6601 040714 012700 000132 MOV #22.*4+2,R0 ;ISSUE CLOCKS UNTIL PHASE ADDRESS 6
6602 040720 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
6603 040726 012762 000040 000026 MOV #DMD,RKMR1(R2)
6604 040734 005300 DEC R0
6605 040736 001370 BNE 1$
6606 040740 005062 000026 CLR RKMR1(R2) ;FINISH COMMAND IN NORMAL MODE
6607 040744 013700 003632 MOV WAITIM,R0 ;WAIT FOR READY
6608 040750 105762 000000 2$: TSTB RKCS1(R2)
6609 040754 100402 BMI 3$
6610 040756 005300 DEC R0
6611 040760 001373 BNE 2$
6612 040762 016237 000000 003440 3$: MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
6613 040770 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
6614 040776 016237 000012 003452 MOV RKDS(R2),T.DS ;STORE DRIVE STATUS REG
6615 041004 016237 000014 003454 MOV RKER(R2),T.ER ;STORE ERROR REG
6616 041012 012737 100224 003500 MOV #CERR!RDY!RDHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6617 041020 012737 000100 003510 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
6618 041026 012737 100001 003512 MOV #SVAL!DRA,E.DS ;LOAD EXPECTED DRIVE STATUS REG
6619 041034 012737 000020 003514 MOV #FMTE,E.ER ;LOAD EXPECTED ERROR REG
6620 041042 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
6621 041050 001401 BEQ 4$ ;:YES, CONTINUE
6622 041052 104175 ERROR 175
6623 041054 023737 003510 003450 4$: CMP E.CS2,T.CS2 ;CHECK COMMAND AND STATU REG 2 CORRECT
    
```

```

6624 041062 001401 BEQ 5$ ;YES, CONTINUE
6625 041064 104176 ERROR 176
6626 041066 023737 003512 003452 5$: CMP E.DS,T.DS ;CHECK IF DRIVE STRATUS REG CORRECT
6627 041074 001401 BEQ 6$ ;YES, CONTINUE
6628 041076 104177 ERROR 177
6629 041100 023737 003514 003454 6$: CMP E.ER,T.ER ;CHECK IF ERR REG CORRECT
6630 041106 001401 BEQ 7$ ;YES, CONTINUE
6631 041110 104200 ERROR 200
6632 041112 013737 003440 003540 7$: MOV T.CS1,P.CS1 ;STORE PREVIOUS CONTENTS OF
6633 041120 013737 003450 003550 MOV T.CS2,P.CS2 ; COMMAND AND STATUS REG 1
6634 041126 013737 003452 003552 MOV T.DS,P.DS ; COMMAND AND STATUS REG 2
6635 041134 013737 003454 003554 MOV T.ER,P.ER ; DRIVE STATUS REG
6636 ; AND ERROR REG
6637 041142 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
6638 041150 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
6639 041156 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
6640 041164 016237 000012 003452 MOV RKDS(R2),T.DS ;STORE DRIVE STATUS REG
6641 041172 016237 000014 003454 MOV RKER(R2),T.ER ;STORE ERROR REG
6642 041200 012737 000200 003500 MOV #RDY,E.CS1 ;LOAD EXPECTED CS1
6643 041206 012737 000100 003510 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
6644 041214 005037 003512 CLR E.DS ;LOAD EXPECTED DRIVE STATUS REG
6645 041220 005037 003514 CLR E.ER ;LOAD EXPECTED ERROR REG
6646 041224 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
6647 041232 001401 BEQ 11$ ;YES, CONTINUE
6648 041234 104211 ERROR 211 ;CS1 INCORRECT
6649 041236 023737 003510 003450 11$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
6650 041244 001401 BEQ 12$ ;YES, CONTINUE
6651 041246 104212 ERROR 212 ;CS2 INCORRECT
6652 041250 023737 003512 003452 12$: CMP E.DS,T.DS ;CHECK DRIVE STATUS CORRECT
6653 041256 001401 BEQ 13$ ;YES, CONTINUE
6654 041260 104213 ERROR 213 ;DRIVE STATUS REG INCORRECT
6655 041262 023737 003514 003454 13$: CMP E.ER,T.ER ;CHECK IF ERROR REG CORRECT
6656 041270 001401 BEQ 14$ ;YES, GO ON TO NEXT TEST
6657 041272 104214 ERROR 214 ;ERROR REG INCORRECT
6658 041274 14$:
6659
6660
6661
6662
6663
6664
6665
6666
6667
6668
6669
6670
6671
6672
6673
6674
6675
6676
6677
6678
6679
  
```

```

*****
*TEST 50          FORMAT ERROR (PART 2)
*
* CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN
* RK06, IN 24 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE 0.
* CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN OFF
* MAINTENANCE MODE. MAKE SURE FORMAT ERROR,
* DRIVE AVAILABLE AND CONTROLLER ERROR SET.
*****
  
```

```

TST50: SCOPE
6671 041274 000004 MOV #100,$TIMES ;DO 100. ITERATIONS
6672 041276 012737 000144 001200 MOV $BASE,R2 ;LOAD RK611 BASE
6673 041304 013702 001270 MOV #SCLR,RKCS2(R2) ;CLEAR RK611 SUBSYSTEM
6674 041310 012762 000040 000010 MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
6675 041316 012762 000040 000026 MOV #3,RKDCY(R2) ;LOAD CYLINDER ADDRESS REG
6676 041324 012762 000003 000020 MOV #RDHEAD!CFMT,RKCS1(R2) ;ISSUE READ HEADER
6677 041332 012762 010025 000000 MOV #22.*4+2,R0 ;ISSUE CLOCKS UNTIL PHASE ADDRESS 6
6678 041340 012700 000132 MOV #DMD!MCLK,RKMR1(R2)
6679 041344 012762 000440 000026 1$: MOV
  
```

```

6680 041352 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6681 041360 005300      DEC      RO
6682 041362 001370      BNE      1$
6683 041364 005062 000026      CLR      RKMR1(R2)      ;FINISH COMMAND IN NORMAL MODE
6684 041370 013700 003632      MOV      WAITIM,RO      ;WAIT FOR READY
6685 041374 105762 000000      2$:     TSTB     RKCS1(R2)
6686 041400 100402      BMI      3$
6687 041402 005300      DEC      RO
6688 041404 001373      BNE      2$
6689 041406 016237 000000 003440 3$:     MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
6690 041414 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
6691 041422 016237 000012 003452      MOV      RKDS(R2),T.DS  ;STORE DRIVE STATUS REG
6692 041430 016237 000014 003454      MOV      RKER(R2),T.ER  ;STORE ERROR REG
6693 041436 012737 110224 003500      MOV      #CERR!RDY!RDHEAD!CFMT<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6694 041444 012737 000100 003510      MOV      #IR,E.CS2      ;LOAD EXPECTED CS2
6695 041452 012737 100001 003512      MOV      #SVAL!DRA,E.DS ;LOAD EXPECTED DRIVE STATUS REG
6696 041460 012737 000030 003514      MOV      #FMTE!DRPAR,E.ER ;LOAD EXPECTED ERROR REG
6697 041466 023737 003500 003440      CMP      E.CS1,T.CS1    ;CHECK COMMAND AND STATUS REG 1 CORRECT
6698 041474 001401      BEQ      4$            ;YES, CONTINUE
6699 041476 104201      ERROR    201
6700 041500 023737 003510 003450 4$:     CMP      E.CS2,T.CS2    ;CHECK COMMAND AND STATU REG 2 CORRECT
6701 041506 001401      BEQ      5$            ;YES, CONTINUE
6702 041510 104202      ERROR    202
6703 041512 023737 003512 003452 5$:     CMP      E.DS,T.DS      ;CHECK IF DRIVE STRATUS REG CORRECT
6704 041520 001401      BEQ      6$            ;YES, CONTINUE
6705 041522 104203      ERROR    203
6706 041524 023737 003514 003454 6$:     CMP      E.ER,T.ER      ;CHECK IF ERR REG CORRECT
6707 041532 001401      BEQ      7$            ;YES, CONTINUE
6708 041534 104204      ERROR    204
6709 041536 013737 003440 003540 7$:     MOV      T.CS1,P.CS1    ;STORE PREVIOUS CONTENTS OF
6710 041544 013737 003450 003550      MOV      T.CS2,P.CS2    ;  COMMAND AND STATUS REG 1
6711 041552 013737 003452 003552      MOV      T.DS,P.DS      ;  COMMAND AND STATUS REG 2
6712 041560 013737 003454 003554      MOV      T.ER,P.ER      ;  DRIVE STATUS REG
6713                                ;  AND ERROR REG
6714 041566 012762 100000 000000      MOV      #CCLR,RKCS1(R2);CLEAR RK611
6715 041574 016237 000000 003440      MOV      RKCS1(R2),T.CS1;STORE COMMAND AND STATUS REG 1
6716 041602 016237 000010 003450      MOV      RKCS2(R2),T.CS2;STORE COMMAND AND STATUS REG 2
6717 041610 016237 000012 003452      MOV      RKDS(R2),T.DS  ;STORE DRIVE STATUS REG
6718 041616 016237 000014 003454      MOV      RKER(R2),T.ER  ;STORE ERROR REG
6719 041624 012737 000200 003500      MOV      #RDY,E.CS1     ;LOAD EXPECTED CS1
6720 041632 012737 000100 003510      MOV      #IR,E.CS2     ;LOAD EXPECTED CS2
6721 041640 005037 003512      CLR      E.DS          ;LOAD EXPECTED DRIVE STATUS REG
6722 041644 005037 003514      CLR      E.ER          ;LOAD EXPECTED ERROR REG
6723 041650 023737 003500 003440      CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
6724 041656 001401      BEQ      11$           ;YES, CONTINUE
6725 041660 104211      ERROR    211           ;CS1 INCORRECT
6726 041662 023737 003510 003450 11$:     CMP      E.CS2,T.CS2    ;CHECK CS2 CORRECT
6727 041670 001401      BEQ      12$           ;YES, CONTINUE
6728 041672 104212      ERROR    212           ;CS2 INCORRECT
6729 041674 023737 003512 003452 12$:     CMP      E.DS,T.DS      ;CHECK DRIVE STATUS CORRECT
6730 041702 001401      BEQ      13$           ;YES, CONTINUE
6731 041704 104213      ERROR    213           ;DRIVE STATUS REG INCORRECT
6732 041706 023737 003514 003454 13$:     CMP      E.ER,T.ER      ;CHECK IF ERROR REG CORRECT
6733 041714 001401      BEQ      14$           ;YES, GO ON TO NEXT TEST
6734 041716 104214      ERROR    214           ;ERROR REG INCORRECT
6735 041720      14$:
  
```

6736  
6737  
6738  
6739  
6740  
6741  
6742  
6743  
6744  
6745  
6746  
6747  
6748 041720 000004  
6749 041722 012737 000144 001200  
6750 041730 013702 001270  
6751 041734 012762 000040 000010  
6752 041742 012762 000040 000026  
6753 041750 012762 000003 000020  
6754 041756 012762 177775 000002  
6755 041764 012762 064630 000004  
6756 041772 012762 000027 000000  
6757 042000 012700 000132  
6758 042004 012762 000440 000026 1\$:  
6759 042012 012762 000040 000026  
6760 042020 005300  
6761 042022 001370  
6762 042024 005062 000026  
6763 042030 013700 003632  
6764 042034 105762 000000 2\$:  
6765 042040 100402  
6766 042042 005300  
6767 042044 001373  
6768 042046 016237 000000 003440 3\$:  
6769 042054 016237 000010 003450  
6770 042062 016237 000012 003452  
6771 042070 016237 000014 003454  
6772 042076 012737 100226 003500  
6773 042104 012737 000100 003510  
6774 042112 012737 100001 003512  
6775 042120 012737 000000 003514  
6776 042126 023737 003500 003440  
6777 042134 001401  
6778 042136 104205  
6779 042140 023737 003510 003450 4\$:  
6780 042146 001401  
6781 042150 104206  
6782 042152 023737 003512 003452 5\$:  
6783 042160 001401  
6784 042162 104207  
6785 042164 023737 003514 003454 6\$:  
6786 042172 001401  
6787 042174 104210  
6788 042176 013737 003440 003540 7\$:  
6789 042204 013737 003450 003550  
6790 042212 013737 003452 003552  
6791 042220 013737 003454 003554

```
*****  
:TEST 51 FAULT SETTING CONTROLLER ERROR  
:  
: CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE  
: CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO  
: AN RK06, IN 26 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE  
: 0. CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6.  
: TURN OFF MAINTENANCE MODE. MAKE SURE DRIVE  
: AVAILABLE AND CONTROLLER ERROR SET.  
:*****  
TST51: SCOPE  
MOV #100, $TIMES ;;DO 100. ITERATIONS  
MOV $BASE, R2 ;LOAD RK611 BASE  
MOV #SCLR, RKCS2(R2) ;CLEAR RK611 SUBSYSTEM  
MOV #DMD, RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE  
MOV #3, RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG  
MOV #-3, RKWC(R2) ;LOAD WORD COUNT  
MOV #WRBUFF, RKBA(R2) ;LOAD BUS ADDRESS  
MOV #WRHEAD, RKCS1(R2) ;ISSUE WRITE HEADER  
MOV #22, R0 ;ISSUE CLOCKS UNTIL PHASE ADDRESS 6  
1$: MOV #DMD!MCLK, RKMR1(R2)  
MOV #DMD, RKMR1(R2)  
DEC R0  
BNE 1$  
CLR RKMR1(R2) ;FINISH COMMAND IN NORMAL MODE  
MOV WAITIM, R0 ;WAIT FOR READY  
2$: TSTB RKCS1(R2)  
BMI 3$  
DEC R0  
BNE 2$  
3$: MOV RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG 1  
MOV RKCS2(R2), T.CS2 ;STORE COMMAND AND STATUS REG 2  
MOV RKDS(R2), T.DS ;STORE DRIVE STATUS REG  
MOV RKER(R2), T.ER ;STORE ERROR REG  
MOV #CERR!RDY!WRHEAD<^C<GO>>, E.CS1 ;LOAD EXPECTED CS1  
MOV #IR, E.CS2 ;LOAD EXPECTED CS2  
MOV #SVAL!DRA, E.DS ;LOAD EXPECTED DRIVE STATUS REG  
MOV #0, E.ER ;LOAD EXPECTED ERROR REG  
CMP E.CS1, T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT  
BEQ 4$ ;YES, CONTINUE  
4$: ERROR 205  
CMP E.CS2, T.CS2 ;CHECK COMMAND AND STATU REG 2 CORRECT  
BEQ 5$ ;YES, CONTINUE  
5$: ERROR 206  
CMP E.DS, T.DS ;CHECK IF DRIVE STRATUS REG CORRECT  
BEQ 6$ ;YES, CONTINUE  
6$: ERROR 207  
CMP E.ER, T.ER ;CHECK IF ERR REG CORRECT  
BEQ 7$ ;YES, CONTINUE  
7$: ERROR 210  
MOV T.CS1, P.CS1 ;STORE PREVIOUS CONTENTS OF  
MOV T.CS2, P.CS2 ; COMMAND AND STATUS REG 1  
MOV T.DS, P.DS ; COMMAND AND STATUS REG 2  
MOV T.ER, P.ER ; DRIVE STATUS REG
```



6816  
6817  
6818  
6819  
6820  
6821  
6822  
6823  
6824  
6825 042360 .  
6826 042360 000004  
6827 042362 005037 001102  
6828 042366 005037 001200  
6829 042372 005237 001222  
6830 042376 042737 100000 001222  
6831 042404 005327  
6832 042406 000001  
6833 042410 003063  
6834 042412 012737  
6835 042414 000001  
6836 042416 042406  
6837 042420 104401 042426  
6838 042424 000407  
6839  
6840 042444  
6841 042444 013746 001222  
6842  
6843 042450 104405  
6844 042452 104401 042460  
6845 042456 000421  
6846  
6847 042522  
6848 042522 013746 001112  
6849  
6850 042526 104405  
6851 042530 104401 001211  
6852 042534 005037 001112  
6853 042540 013700 000042  
6854 042544 001405  
6855 042546 000005  
6856 042550 004710  
6857 042552 000240  
6858 042554 000240  
6859 042556 000240  
6860 042560  
6861 042560 000137  
6862 042562 004636  
6863 042564 377 377 000  
6864 042570  
6865  
6866  
6867  
6868  
6869 042570 052737 002400 003524  
6870 042576 012762 000440 000026  
6871 042604 016237 000026 003464

.SBTTL END OF PASS ROUTINE

```
::*****  
:*INCREMENT THE PASS NUMBER ($PASS)  
:*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY'  
:*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS  
:*IF THERES A MONITOR GO TO IT  
:*IF THERE ISN'T JUMP TO NEWPAS
```

\$EOP:

```
SCOPE  
CLR $STNM ;;ZERO THE TEST NUMBER  
CLR $TIMES ;;ZERO THE NUMBER OF ITERATIONS  
INC $PASS ;;INCREMENT THE PASS NUMBER  
BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER  
DEC (PC)+ ;;LOOP?  
$EOPCT: .WORD 1  
BGT $DOAGN ;;YES  
MOV (PC)+,@(PC)+ ;;RESTORE COUNTER  
$ENDCT: .WORD 1  
TYPE ,65$ ;;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 $ERTTL FOR TYPEOUT  
TYPDS ;;TOTAL NUMBER OF ERRORS  
TYPE ,67$ ;;GO TYPE--DECIMAL ASCII WITH SIGN  
CLR $ERTTL ;;TYPE CARRIAGE RETURN, LINE FEED  
$GET42: MOV @#42,R0 ;;CLEAR ERROR TOTAL  
BEQ $DOAGN ;;GET MONITOR ADDRESS  
RESET ;;BRANCH IF NO MONITOR  
$ENDAD: JSR PC,(R0) ;;CLEAR THE WORLD  
NOP ;;GO TO MONITOR  
NOP ;;SAVE ROOM  
NOP ;;FOR  
;;ACT11  
$DOAGN: JMP @(PC)+ ;;RETURN  
$RTNAD: .WORD NEWPAS  
$ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING  
.EVEN
```

.SBTTL SIMULATE ONE BIT OF WRITE DATA IN MAINTANENCE MODE

```
WRTBIT: BIS #MCLK!MEWD,E.MR1 ;CREATE EXPECTED MAINT. REG. 1  
MOV #DMD!MCLK,RKMR1(R2) ;PROVIDED 1ST UPWARD TRANSITION  
MOV RKMR1(R2),T.MR1 ;STORE MAINT. REG. 1
```

6872	042612	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK IF MAINT REG 1 CORRECT
6873	042620	001416				BEQ	3\$	:YES, PROVIDE DOWNWARD TRANSITION
6874	042622	012737	042642	001202		MOV	#1\$, \$ESCAPE	:LOAD ESCAPE FOR LOOP ON ERROR
6875	042630	012737	063747	003172		MOV	#EMW1,EMW+2	:LOAD ERROR MESSAGE
6876	042636	011646				MOV	(SP),-(SP)	:SAVE RETURN
6877	042640	000207				RTS	PC	:MR1 INCORRECT ON UPWARD TRANSITION
6878								
6879	042642	032777	001000	136270	1\$:	BIT	#SW9,@SWR	:CHECK IF LOOP ON ERROR
6880	042650	001402				BEQ	3\$	:NO, CONTINUE
6881	042652	000137	043524			JMP	63\$	:YES, LOOP ON ERROR
6882								
6883	042656	042737	014400	003524	3\$:	BIC	#MCLK!PCA!PCD,E.MR1	:INITIALIZE MAINTENANCE REG. 1
6884	042664	052737	042000	003524		BIS	#MEWD!WRTGAT,E.MR1	
6885	042672	005737	003614			TST	PR.BIT	:CHECK IF ONE
6886	042676	001152				BNE	20\$	:YES, SIMULATE ONE
6887	042700	005737	003616			TST	M1.BIT	:CHECK IF PREVIOUS ONE
6888	042704	001023				BNE	10\$	:YES, NO TRANSITION
6889	042706	042737	002000	003524		BIC	#MEWD,E.MR1	:INDICATE TRANSITION
6890	042714	005737	003612			TST	P1.BIT	:CHECK IF NEXT BIT = 1
6891	042720	001007				BNE	5\$	:YES, CHECK FOR PRECOMP ADVANCE
6892	042722	005737	003620			TST	M2.BIT	:CHECK FOR PRECOMP. ADVANCE
6893	042726	001412				BEQ	10\$	:NO, CLOCK IN ZERO
6894	042730	052737	010000	003524		BIS	#PCD,E.MR1	:SET PRECOMP. DELAY
6895	042736	000406				BR	10\$	:CLOCK IN ZERO
6896								
6897	042740	005737	003620		5\$:	TST	M2.BIT	:CHECK FOR PRECOMP. ADVANCE
6898	042744	001003				BNE	10\$	:CLOCK IN ZERO
6899	042746	052737	004000	003524		BIS	#PCA,E.MR1	:SET PRECOMP. ADVANCE
6900	042754	012762	000040	000026	10\$:	MOV	#DMD,RKMR1(R2)	:CLOCK IN DATA BIT
6901	042762	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MR1
6902	042770	023737	003464	003524		CMP	T.MR1,E.MR1	:CHECK IF MR1 CORRECT
6903	042776	001416				BEQ	12\$	:YES, CONTINUE
6904	043000	012737	043020	001202		MOV	#11\$, \$ESCAPE	:LOAD ESCAPE FOR LOOP ON ERROR
6905	043006	012737	064036	003172		MOV	#EMW2,EMW+2	:LOAD ERROR MESSAGE
6906	043014	011646				MOV	(SP),-(SP)	:SAVE RETURN
6907	043016	000207				RTS	PC	:MR1 INCORRECT
6908								
6909	043020	032777	001000	136112	11\$:	BIT	#SW9,@SWR	:CHECK IF LOOP ON ERROR
6910	043026	001402				BEQ	12\$	:NO, CONTINUE
6911	043030	000137	043524			JMP	63\$	:YES, LOOP ON ERROR
6912								
6913	043034	052737	002400	003524	12\$:	BIS	#MCLK!MEWD,E.MR1	:CREATE EXPECTED MAINT REG 1
6914	043042	012762	000440	000026		MOV	#DMD!MCLK,RKMR1(R2)	:PROVIDE 2ND UPWARD TRANSITION
6915	043050	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:STORE MAINT REG. 1
6916	043056	023737	003524	003464		CMP	E.MR1,T.MR1	:CHECK IF MAINT REG. 1 CORRECT
6917	043064	001416				BEQ	15\$	:YES, CONTINUE
6918	043066	012737	043106	001202		MOV	#13\$, \$ESCAPE	:LOAD ESCAPE FOR LOOP ON ERROR
6919	043074	012737	064127	003172		MOV	#EMW3,EMW+2	:LOAD ERROR MESSAGE
6920	043102	011646				MOV	(SP),-(SP)	:SAVE RETURN
6921	043104	000207				RTS	PC	:MR1 INCORRECT
6922								
6923	043106	032777	001000	136024	13\$:	BIT	#SW9,@SWR	:CHECK IF LOOP ON ERROR
6924	043114	001402				BEQ	15\$	:NO, CONTINUE
6925	043116	000137	043524			JMP	63\$	:YES, LOOP ON ERROR
6926								
6927	043122	052737	002000	003524	15\$:	BIS	#MEWD,E.MR1	:RESET TRANSITION INDICATION



```

6928 043130 042737 000400 003524      BIC      #MCLK,E.MR1
6929 043136 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;SUPPLY LAST PART OF DATA
6930 043144 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MR1
6931 043152 023737 003464 003524      CMP      T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
6932 043160 001414      BEQ      18$ ;YES, RETURN
6933 043162 012737 043202 001202      MOV      #17$, $ESCAPE ;LOAD ESCAPE FOR LOOP ON ERROR
6934 043170 012737 064216 003172      MOV      #EMW4,EMW+2 ;LOAD ERROR MESSAGE
6935 043176 011646      MOV      (SP),-(SP) ;SAVE RETURN
6936 043200 000207      RTS      PC ;MR1 INCORRECT
6937
6938 043202 032777 001000 135730 17$: BIT      #SW9,@SWR ;CHECK IF LOOP ON ERROR
6939 043210 001145      BNE      63$ ;YES, LOOP ON ERROR
6940 043212 005037 001202      CLR      $ESCAPE ;CLEAR ESCAPE
6941 043216 062716 000002      ADD      #2,(SP) ;ADJUST RETURN
6942 043222 000207      RTS      PC ;RETURN
6943
6944 043224 005737 003612      20$: TST      P1.BIT ;CHECK IN NEXT BIT A ONE
6945 043230 001007      BNE      30$ ;YES, CHECK IF PRECOMP DELAY
6946 043232 005737 003616      TST      M1.BIT ;CHECK FOR PRECOMP ADVANCE
6947 043236 001415      BEQ      40$ ;NO, CLOCK IN DATA BIT
6948 043240 052737 004000 003524      BIS      #PCA,E.MR1 ;SET PRECOMP. ADVANCE
6949 043246 000411      BR       40$ ;CHECK MR1
6950
6951 043250 042737 000400 003524 30$: BIC      #MCLK,E.MR1 ;RESET MAINT CLOCK IN EXPECTED MR1
6952 043256 005737 003616      TST      M1.BIT ;CHECK FOR PRECOMP DELAY
6953 043262 001003      BNE      40$ ;NO, CHECK MR1
6954 043264 052737 010000 003524      BIS      #PCD,E.MR1 ;SET SET PRECOMP DELAY
6955 043272 012762 000040 000026 40$: MOV      #DMD,RKMR1(R2) ;CLOCK IN DATA BIT
6956 043300 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MR1
6957 043306 023737 003464 003524      CMP      T.MR1,E.MR1 ;CHECK MR1 CORRECT
6958 043314 001414      BEQ      42$ ;YES, CLOCK IN REST OF BIT
6959 043316 012737 043336 001202      MOV      #41$, $ESCAPE ;LOAD ESCAPE FOR LOOP ON ERROR
6960 043324 012737 064036 003172      MOV      #EMW2,EMW+2 ;LOAD ERROR MESSAGE
6961 043332 011646      MOV      (SP),-(SP) ;SAVE RETURN
6962 043334 000207      RTS      PC ;MR1 INCORRECT
6963
6964 043336 032777 001000 135574 41$: BIT      #SW9,@SWR ;CHECK IF LOOP ON ERROR
6965 043344 001067      BNE      63$ ;YES, LOOP ON ERROR
6966 043346 052737 000400 003524 42$: BIS      #MCLK,E.MR1 ;CHREATE EXPECTED MAINT. REG. 1
6967 043354 012762 000440 000026      MOV      #DMD!MCLK,RKMR1(R2) ;PROVIDE 2ND UPWARD TRANSITION
6968 043362 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MAINT REG 1
6969 043370 023737 003524 003464      CMP      E.MR1,T.MR1 ;CHECK IF MAINT REG 1 CORRECT
6970 043376 001414      BEQ      45$ ;YES, CONTINUE
6971 043400 012737 043420 001202      MOV      #43$, $ESCAPE ;LOAD ESCAPE
6972 043406 012737 064127 003172      MOV      #EMW3,EMW+2 ;LOAD ERROR MESSAGE
6973 043414 011646      MOV      (SP),-(SP) ;SAVE RETURN
6974 043416 000207      RTS      PC ;MR1 INCORRECT
6975
6976 043420 032777 001000 135512 43$: BIT      #SW9,@SWR ;CHECK IF LOOP ON ERROR
6977 043426 001036      BNE      63$ ;YES, LOOP ON ERROR
6978 043430 042737 002400 003524 45$: BIC      #MEWD!MCLK,E.MR1 ;SET TRANSITION
6979 043436 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;CLOCK TRANSITION
6980 043444 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MR1
6981 043452 023737 003464 003524      CMP      T.MR1,E.MR1 ;CHECK IF MR1 CORRECT
6982 043460 001414      BEQ      50$ ;YES, RETURN
6983 043462 012737 043502 001202      MOV      #47$, $ESCAPE ;LOAD ESCAPE FOR LOOP ON ERROR
  
```

```

6984 043470 012737 064216 003172      MOV      #EMW4,EMW+2      ;LOAD ERROR MESSAGE
6985 043476 011646                MOV      (SP),-(SP)      ;SAVE RETURN
6986 043500 000207                RTS      PC              ;MR1 INCORRECT
6987
6988 043502 032777 001000 135430 47$:  BIT      #SW9,@SWR      ;CHECK IF LOOP ON ERROR
6989 043510 001005                BNE     63$              ;YES, LOOP ON ERROR
6990 043512 005037 001202          50$:  CLR      $ESCAPE      ;CLEAR ESCAPE
6991 043516 062716 000002          ADD     #2,(SP)         ;ADJUST RETURN
6992 043522 000207                RTS      PC              ;RETURN
6993
6994 043524 005037 001202          63$:  CLR      $ESCAPE      ;CLEAR ESCAPE
6995 043530 012706 001100          MOV     #STACK,SP      ;FORCE STACK
6996 043534 000177 135350          JMP     @SLPERR        ;LOOP ON ERROR
6997
6998      .SBTTL  SIMULATE ONE BIT OR READ DATA IN MAINTANENCE MODE
6999
7000 043540 005737 003614          RDBIT: TST     PR.BIT      ;CHECK IF ONE
7001 043544 001024                BNE     10$             ;YES, SIMULATE ONE
7002 043546 005737 003616          TST     M1.BIT        ;CHECK IF PREVIOUS ONE
7003 043552 001404                BEQ     4$              ;NO, INSERT TRANSITION
7004 043554 012762 000440 000026          MOV     #DMD!MCLK,RKMR1(R2) ;YES, DO NOT INSERT TRANSITION
7005 043562 000403                BR      5$              ;CLOCK IN ZERO
7006
7007 043564 012762 001440 000026 4$:  MOV     #DMD!MCLK!MERD,RKMR1(R2) ;INSERT TRANSITION
7008 043572 012762 000040 000026 5$:  MOV     #DMD,RKMR1(R2) ;CLOCK IN ZERO
7009 043600 012762 000440 000026          MOV     #DMD!MCLK,RKMR1(R2)
7010 043606 012762 000040 000026          MOV     #DMD,RKMR1(R2)
7011 043614 000207                RTS      PC              ;RETURN
7012
7013 043616 012762 000440 000026 10$: MOV     #DMD!MCLK,RKMR1(R2) ;CLOCK IN ONE
7014 043624 012762 000040 000026          MOV     #DMD,RKMR1(R2)
7015 043632 012762 001440 000026          MOV     #DMD!MCLK!MERD,RKMR1(R2)
7016 043640 012762 000040 000026          MOV     #DMD,RKMR1(R2)
7017 043646 000207                RTS      PC              ;RETURN
7018
7019      .SBTTL  MEMORY CHECK ENABLE TRAP
7020
7021 043650 012737 043664 001202 MEMERR: MOV     #10$, $ESCAPE ;LOAD ESCAPE
7022 043656 011637 003604                MOV     (SP),TRAPP      ;STORE PC
7023 043662 104147                ERROR   147            ;REPORT MEM PARITY ERROR
7024 043664 005037 001202          10$:  CLR      $ESCAPE      ;CLEAR ESCAPE
7025 043670 032777 001000 135242          BIT     #SW9,@SWR      ;CHECK IF LOOP ON ERROR
7026 043676 001001                BNE     15$            ;YES, FORCE STACK AND TRY AGAIN
7027 043700 000002                RTI     ;NO, RETURN
7028
7029 043702 012706 001100          15$:  MOV     #STACK,SP      ;INITIALIZE STACK
7030 043706 000177 135176          JMP     @SLPERR        ;LOOP ON ERROR
7031
7032      .SBTTL  ROUTINE TO SIZE MEMORY
7033
7034      ;*****
7035      ;*CALL:
7036      ;*      JSR     PC,$SIZE
7037      ;*      RETURN
7038      ;*SLSTAD WILL CONTAIN:
7039      ;*      WITH KT11 OPTION
  
```

-- LAST VIRTUAL ADDRESS OF THE LAST BANK

```
7040          : * WITHOUT KT11 OPTION      -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
7041          : * $LSTBK WILL CONTAIN THE LAST BANK AS A SAF
7042          : * $SKT11 IS THE MEMORY MANAGEMENT KEY
7043          : * $BIT07 = 0 DON'T USE MEMORY MANAGEMENT
7044          : * MUST BE SETUP BEFORE THE CALL
7045          : * $BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
7046          : * DETERMINED BY ROUTINE
7047
7048 043712 010046 $SIZE: MOV R0,-(SP)      ;;SAVE R0 ON THE STACK
7049 043714 010146      MOV R1,-(SP)      ;;SAVE R1 ON THE STACK
7050 043716 010246      MOV R2,-(SP)      ;;SAVE R2 ON THE STACK
7051 043720 010346      MOV R3,-(SP)      ;;SAVE R3 ON THE STACK
7052 043722 013746 000004  MOV @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
7053 043726 013746 000006  MOV @#ERRVEC+2,-(SP)
7054 043732 010600      MOV SP,R0      ;;SAVE THE STACK POINTER
7055          ;;SET THE ERRVEC PS TO THE PRESENT PS
7056 043734 104400      TRAP      ;;PUSH OLD PSW AND PC ON STACK
7057 043736 012637 000006  MOV (SP)+,@#ERRVEC+2  ;;SAVE THE PSW IN @#ERRVEC+2
7058 043742 012701 003776  MOV #3776,R1      ;;SETUP ADDRESS
7059 043746 105727      TSTB (PC)+      ;;USE MEMORY MANAGEMENT?
7060 043750 000200 $KT11: .WORD 200      ;;SET TO USE MEMORY MANAGEMENT
7061 043752 100062      BPL SCORE      ;;BR IF NO
7062 043754 012737 044112 000004  MOV #SKTNEX,@#ERRVEC  ;;SET FOR TIMEOUT
7063 043762 005737 177572      TST @#SR0      ;;KT11 ARE YOU THERE?
7064 043766 052737 100000 043750  BIS #100000,$KT11  ;;YES--SET KT11 KEY
7065 043774 005046      CLR -(SP)      ;;INITIALIZE FOR 'PAR' LOADING
7066 043776 012702 172340      MOV #KIPAR0,R2      ;;ADDRESS OF FIRST 'PAR'
7067 044002 012703 000010      MOV #^DB,R3      ;;LOAD EIGHT 'PAR.'S' AND EIGHT 'PDR.'S'
7068 044006 012762 077406 177740 1$: MOV #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
7069 044014 011622      MOV (SP),(R2)+    ;;LOAD 'PAR'
7070 044016 062716 000200      ADD #200,(SP)     ;;UPDATE FOR NEXT 'PAR'
7071 044022 077307      SOB R3,1$      ;;LOOP UNTIL ALL EIGHT ARE LOADED
7072 044024 012742 177600      MOV #177600,-(R2)  ;;SETUP KIPAR7 FOR I/O
7073 044030 005042      CLR -(R2)      ;;SETUP KIPAR6 FOR TESTING
7074 044032 012737 044050 000004  MOV #2$,@#ERRVEC  ;;CATCH TIMEOUT IF NO SR3
7075 044040 012737 000020 172516  MOV #20,@#SR3     ;;ENABLE 22 BIT MODE
7076 044046 000401      BR 3$         ;;THIS PDP-11 HAS A SR3 REGISTER
7077 044050 022626      2$: CMP (SP)+,(SP)+  ;;CLEAN OFF THE STACK--NO SR3
7078 044052 005237 177572      3$: INC @#SR0      ;;TURN ON MEMORY MANAGEMENT
7079 044056 012737 044102 000004  MOV #SKTOUT,@#ERRVEC  ;;SET FOR TIME OUT
7080 044064 005737 143776      4$: TST @#143776     ;;TRAP ON NON-EX-MEM
7081 044070 062712 000040      ADD #40,(R2)     ;;MAKE A 1K STEP
7082 044074 023712 172356      CMP @#KIPAR7,(R2)  ;;LAST ONE?
7083 044100 101371      BHI 4$        ;;NO--TRY IT
7084 044102 011202      SKTOUT: MOV (R2),R2  ;;GET LAST BANK+1
7085 044104 005037 177572      CLR @#SR0      ;;TURN OFF MEMORY MANAGEMENT
7086 044110 000421      BR $SIZEX
7087 044112 042737 100000 043750  SKTNEX: BIC #100000,$KT11  ;;KT11 NON-EXISTENT
7088 044120 012737 044150 000004  SCORE: MOV #SCROUT,@#ERRVEC  ;;SET FOR TIMEOUT
7089 044126 005002      CLR R2        ;;SET UP BANK
7090 044130 062701 004000      1$: ADD #4000,R1    ;;INCREMENT BY 1K
7091 044134 062702 000040      ADD #40,R2     ;;1K STEP
7092 044140 005711      TST (R1)      ;;TRAP ON TIME OUT
7093 044142 022701 177776      CMP #177776,R1  ;;LAST ONE
7094 044146 001370      BNE 1$       ;;NO--TRY AGAIN
7095 044150 162701 004000  SCROUT: SUB #4000,R1
```



7152	044340	005726		8\$:	TST	(SP)+	:: CLEAN THE BAD TEST NUMBER OFF OF THE STACK	
7153	044342	105737	001103	2\$:	TSTB	\$ERFLG	:: HAS AN ERROR OCCURRED?	
7154	044346	001421			BEQ	3\$	:: BR IF NO	
7155	044350	123737	001115	001103	CMPB	\$ERMAX,\$ERFLG	:: MAX. ERRORS FOR THIS TEST OCCURRED?	
7156	044356	101015			BHI	3\$	:: BR IF NO	
7157	044360	032777	001000	134552	BIT	#BIT09,@SWR	:: LOOP ON ERROR?	
7158	044366	001404			BEQ	4\$	:: BR IF NO	
7159	044370	013737	001110	001106	7\$:	MOV	\$LPERR,\$LPADR	:: SET LOOP ADDRESS TO LAST SCOPE
7160	044376	000446			BR	\$OVER		
7161	044400	105037	001103		4\$:	CLRB	\$ERFLG	:: ZERO THE ERROR FLAG
7162	044404	005037	001200		CLR	\$TIMES	:: CLEAR THE NUMBER OF ITERATIONS TO MAKE	
7163	044410	000415			BR	1\$	:: ESCAPE TO THE NEXT TEST	
7164	044412	032777	004000	134520	3\$:	BIT	#BIT11,@SWR	:: INHIBIT ITERATIONS?
7165	044420	001011			BNE	1\$	:: BR IF YES	
7166	044422	005737	001222		TST	\$PASS	:: IF FIRST PASS OF PROGRAM	
7167	044426	001406			BEQ	1\$	:: INHIBIT ITERATIONS	
7168	044430	005237	001104		INC	\$ICNT	:: INCREMENT ITERATION COUNT	
7169	044434	023737	001200	001104	CMP	\$TIMES,\$ICNT	:: CHECK THE NUMBER OF ITERATIONS MADE	
7170	044442	002024			BGE	\$OVER	:: BR IF MORE ITERATION REQUIRED	
7171	044444	012737	000001	001104	1\$:	MOV	#1,\$ICNT	:: REINITIALIZE THE ITERATION COUNTER
7172	044452	013737	044530	001200	MOV	\$MXCNT,\$TIMES	:: SET NUMBER OF ITERATIONS TO DO	
7173	044460	105237	001102		\$SVLAD:	INCB	\$STNM	:: COUNT TEST NUMBERS
7174	044464	113737	001102	001220	MOVB	\$STNM,\$TESTN	:: SET TEST NUMBER IN APT MAILBOX	
7175	044472	011637	001106		MOV	(SP),\$LPADR	:: SAVE SCOPE LOOP ADDRESS	
7176	044476	011637	001110		MOV	(SP),\$LPERR	:: SAVE ERROR LOOP ADDRESS	
7177	044502	005037	001202		CLR	\$ESCAPE	:: CLEAR THE ESCAPE FROM ERROR ADDRESS	
7178	044506	112737	000001	001115	MOVB	#1,\$ERMAX	:: ONLY ALLOW ONE(1) ERROR ON NEXT TEST	
7179	044514	013777	001102	134420	\$OVER:	MOV	\$STNM,@DISPLAY	:: DISPLAY TEST NUMBER
7180	044522	013716	001106		MOV	\$LPADR,(SP)	:: FUDGE RETURN ADDRESS	
7181	044526	000002			RTI		:: FIXES PS	
7182	044530	003720			\$MXCNT:	2000.	:: MAX. NUMBER OF ITERATIONS	
7183	044532				\$SW08TBL:			
7184	044532	004652			.WORD	TST1+2	:: STARTING ADDRESS OF TEST 1	
7185	044534	005070			.WORD	TST2+2	:: STARTING ADDRESS OF TEST 2	
7186	044536	005306			.WORD	TST3+2	:: STARTING ADDRESS OF TEST 3	
7187	044540	005522			.WORD	TST4+2	:: STARTING ADDRESS OF TEST 4	
7188	044542	005736			.WORD	TST5+2	:: STARTING ADDRESS OF TEST 5	
7189	044544	006420			.WORD	TST6+2	:: STARTING ADDRESS OF TEST 6	
7190	044546	007046			.WORD	TST7+2	:: STARTING ADDRESS OF TEST 7	
7191	044550	007434			.WORD	TST10+2	:: STARTING ADDRESS OF TEST 10	
7192	044552	010560			.WORD	TST11+2	:: STARTING ADDRESS OF TEST 11	
7193	044554	011246			.WORD	TST12+2	:: STARTING ADDRESS OF TEST 12	
7194	044556	012132			.WORD	TST13+2	:: STARTING ADDRESS OF TEST 13	
7195	044560	013020			.WORD	TST14+2	:: STARTING ADDRESS OF TEST 14	
7196	044562	014224			.WORD	TST15+2	:: STARTING ADDRESS OF TEST 15	
7197	044564	015102			.WORD	TST16+2	:: STARTING ADDRESS OF TEST 16	
7198	044566	015540			.WORD	TST17+2	:: STARTING ADDRESS OF TEST 17	
7199	044570	016654			.WORD	TST20+2	:: STARTING ADDRESS OF TEST 20	
7200	044572	017770			.WORD	TST21+2	:: STARTING ADDRESS OF TEST 21	
7201	044574	021104			.WORD	TST22+2	:: STARTING ADDRESS OF TEST 22	
7202	044576	021774			.WORD	TST23+2	:: STARTING ADDRESS OF TEST 23	
7203	044600	022526			.WORD	TST24+2	:: STARTING ADDRESS OF TEST 24	
7204	044602	023260			.WORD	TST25+2	:: STARTING ADDRESS OF TEST 25	
7205	044604	024012			.WORD	TST26+2	:: STARTING ADDRESS OF TEST 26	
7206	044606	024544			.WORD	TST27+2	:: STARTING ADDRESS OF TEST 27	
7207	044610	025276			.WORD	TST30+2	:: STARTING ADDRESS OF TEST 30	

```
7208 044612 026030 .WORD TST31+2 :: STARTING ADDRESS OF TEST 31
7209 044614 026330 .WORD TST32+2 :: STARTING ADDRESS OF TEST 32
7210 044616 027076 .WORD TST33+2 :: STARTING ADDRESS OF TEST 33
7211 044620 027466 .WORD TST34+2 :: STARTING ADDRESS OF TEST 34
7212 044622 030312 .WORD TST35+2 :: STARTING ADDRESS OF TEST 35
7213 044624 031136 .WORD TST36+2 :: STARTING ADDRESS OF TEST 36
7214 044626 031762 .WORD TST37+2 :: STARTING ADDRESS OF TEST 37
7215 044630 032606 .WORD TST40+2 :: STARTING ADDRESS OF TEST 40
7216 044632 033432 .WORD TST41+2 :: STARTING ADDRESS OF TEST 41
7217 044634 034256 .WORD TST42+2 :: STARTING ADDRESS OF TEST 42
7218 044636 035102 .WORD TST43+2 :: STARTING ADDRESS OF TEST 43
7219 044640 035742 .WORD TST44+2 :: STARTING ADDRESS OF TEST 44
7220 044642 036766 .WORD TST45+2 :: STARTING ADDRESS OF TEST 45
7221 044644 040012 .WORD TST46+2 :: STARTING ADDRESS OF TEST 46
7222 044646 040652 .WORD TST47+2 :: STARTING ADDRESS OF TEST 47
7223 044650 041276 .WORD TST50+2 :: STARTING ADDRESS OF TEST 50
7224 044652 041722 .WORD TST51+2 :: STARTING ADDRESS OF TEST 51
7225 *****
7226 .SBTTL LOOP ON INTERNAL ERROR
7227
7228 044654 032777 001000 134256 SCOP1$: BIT #SW9,@SWR ;CHECK IF LOOP ON ERROR
7229 044662 001405 BEQ 5$ ;NO RETURN
7230 044664 105737 001103 TSTB $ERFLG ;CHECK IF ERROR OCCURED
7231 044670 001402 BEQ 5$ ;NO, RETURN
7232 044672 013716 001110 MOV $LPERR,(SP) ;GO BACK TO BEGINNING OF LOOP
7233 044676 000002 5$: RTI ;RETURN
7234 .SBTTL APT COMMUNICATIONS ROUTINE
7235 *****
7236
7237 044700 112737 000001 045144 $ATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
7238 044706 112737 000001 045142 $ATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
7239 044714 000403 BR $ATYC
7240 044716 112737 000001 045144 $ATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
7241 044724 $ATYC:
7242 044724 010046 MOV RO,-(SP) ;;PUSH RO ON STACK
7243 044726 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
7244 044730 105737 045142 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
7245 044734 001450 BEQ 5$ ;;IF NOT: BR
7246 044736 122737 000001 001234 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
7247 044744 001031 BNE 3$ ;;IF NOT: BR
7248 044746 132737 000100 001235 BITB #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
7249 044754 001425 BEQ 3$ ;;IF NOT: BR
7250 044756 017600 000004 MOV @4(SP),RO ;;GET MESSAGE ADDR.
7251 044762 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
7252 044770 005737 001214 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
7253 044774 001375 BNE 1$ ;;IF NOT: WAIT
7254 044776 010037 001230 MOV RO,$MSGAD ;;PUT ADDR IN MAILBOX
7255 045002 105720 2$: TSTB (RO)+ ;;FIND END OF MESSAGE
7256 045004 001376 BNE 2$
7257 045006 163700 001230 SUB $MSGAD,RO ;;SUB START OF MESSAGE
7258 045012 006200 ASR RO ;;GET MESSAGE LNGTH IN WORDS
7259 045014 010037 001232 MOV RO,$MSGGLT ;;PUT LENGTH IN MAILBOX
7260 045020 012737 000004 001214 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
7261 045026 000413 BR 5$
7262 045030 017637 000004 045054 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
7263 045036 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
```

```
7264 045044 013746 177776          MOV    177776,-(SP)      ;;PUSH 177776 ON STACK
7265 045050 004737 045706          JSR    PC,$TYPE        ;;CALL TYPE MACRO
7266 045054 000000                    4$:    .WORD    0
7267 045056                    5$:
7268 045056 105737 045144                    10$:   TSTB   $FFLG          ;;SHOULD REPORT FATAL ERROR?
7269 045062 001416                    BEQ    12$              ;;IF NOT: BR
7270 045064 005737 001234                    TST   $ENV             ;;RUNNING UNDER APT?
7271 045070 001413                    BEQ    12$              ;;IF NOT: BR
7272 045072 005737 001214                    11$:   TST   $MSGTYPE      ;;FINISHED LAST MESSAGE?
7273 045076 001375                    BNE    11$              ;;IF NOT: WAIT
7274 045100 017637 000004 001216    MOV    @4(SP),$FATAL    ;;GET ERROR #
7275 045106 062766 000002 000004    ADD    #2,4(SP)         ;;BUMP RETURN ADDR.
7276 045114 005237 001214                    INC   $MSGTYPE         ;;TELL APT TO TAKE ERROR
7277 045120 105037 045144                    12$:   CLRB   $FFLG          ;;CLEAR FATAL FLAG
7278 045124 105037 045143                    CLRB   $LFLG           ;;CLEAR LOG FLAG
7279 045130 105037 045142                    CLRB   $MFLG           ;;CLEAR MESSAGE FLAG
7280 045134 012601                    MOV    (SP)+,R1         ;;POP STACK INTO R1
7281 045136 012600                    MOV    (SP)+,R0         ;;POP STACK INTO R0
7282 045140 000207                    RTS    PC                ;;RETURN
7283 045142 000          $MFLG: .BYTE    0        ;;MESSG. FLAG
7284 045143 000          $LFLG: .BYTE    0        ;;LOG FLAG
7285 045144 000          $FFLG: .BYTE    0        ;;FATAL FLAG
7286 045146                    .EVEN
7287 000200          APTSIZE=200
7288 000001          APTENV=001
7289 000100          APTSPool=100
7290 000040          APTCSUP=040
7291                    .SBTTL  ERROR HANDLER ROUTINE
7292
7293                    ;;*****
7294                    ;;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
7295                    ;;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
7296                    ;;*AND GO TO TYPERR ON ERROR
7297                    ;;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7298                    ;;*SW15=1      HALT ON ERROR
7299                    ;;*SW13=1      INHIBIT ERROR TYPEOUTS
7300                    ;;*SW10=1     BELL ON ERROR
7301                    ;;*SW09=1     LOOP ON ERROR
7302                    ;;*CALL
7303                    ;;*      ERROR  N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
7304
7305                    $ERROR:
7306 045146 104407                    CKSWR                    ;;TEST FOR CHANGE IN SOFT-SWR
7307 045150 105237 001103                    7$:    INCB   $ERFLG      ;;SET THE ERROR FLAG
7308 045154 001775                    BEQ    7$                ;;DON'T LET THE FLAG GO TO ZERO
7309 045156 013777 001102 133756    MOV    $TSTNM,@DISPLAY  ;;DISPLAY TEST NUMBER AND ERROR FLAG
7310 045164 032777 002000 133746    BIT    #BIT10,@SWR      ;;BELL ON ERROR?
7311 045172 001402                    BEQ    1$                ;;NO - SKIP
7312 045174 104401 001204                    TYPE   $BELL            ;;RING BELL
7313 045200 005237 001112                    1$:    INC   $ERTTL        ;;COUNT THE NUMBER OF ERRORS
7314 045204 011637 001116                    MOV    (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
7315 045210 162737 000002 001116    SUB    #2,$ERRPC
7316 045216 117737 133674 001114    MOVB  @ERRPC,$ITEMB     ;;STRIP AND SAVE THE ERROR ITEM CODE
7317 045224 032777 020000 133706    BIT    #BIT13,@SWR      ;;SKIP TYPEOUT IF SET
7318 045232 001004                    BNE    20$              ;;SKIP TYPEOUTS
7319 045234 004737 045346                    JSR    PC,TYPERR        ;;GO TO USER ERROR ROUTINE
```

```

7320 045240 104401 001211          TYPE      ,SCRLF
7321 045244          20$:          CMPB      #APTENV,$ENV      ;;RUNNING IN APT MODE
7322 045244 122737 000001 001234          BNE       2$          ;;NO,SKIP APT ERROR REPORT
7323 045252 001007          MOVB      $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
7324 045254 113737 001114 045266          JSR       PC,$ATY4      ;;REPORT FATAL ERROR TO APT
7325 045262 004737 044716          .BYTE    0
7326 045266          21$:          .BYTE    0
7327 045267          22$:          BR        22$          ;;APT ERROR LOOP
7328 045270 000777          2$:          TST      @SWR          ;;HALT ON ERROR
7329 045272 005777 133642          BPL      3$          ;;SKIP IF CONTINUE
7330 045276 100002          HALT
7331 045300 000000          CKSWR
7332 045302 104407          3$:          BIT      #BIT09,@SWR      ;;TEST FOR CHANGE IN SOFT-SWR
7333 045304 032777 001000 133626          BEQ      4$          ;;LOOP ON ERROR SWITCH SET?
7334 045312 001402          MOV      $LPERR,(SP)    ;;BR IF NO
7335 045314 013716 001110          TST      $ESCAPE        ;;FUDGE RETURN FOR LOOPING
7336 045320 005737 001202          4$:          BEQ      5$          ;;CHECK FOR AN ESCAPE ADDRESS
7337 045324 001402          MOV      $ESCAPE,(SP)  ;;BR IF NONE
7338 045326 013716 001202          5$:          CMP      #SENDAD,@#42   ;;FUDGE RETURN ADDRESS FOR ESCAPE
7339 045332          6$:          BNE      6$          ;;ACT-11 AUTO-ACCEPT?
7340 045332 022737 042550 000042          HALT
7341 045340 001001          RTI          ;;BRANCH IF NO
7342 045342 000000          ;;YES
7343 045344          6$:          RTI          ;;RETURN
7344 045344 000002
7345
7346
7347
7348
7349
7350
7351
7352
7353
7354
7355

```

```

*****
.SBTTL TYPE ERROR ROUTINE
*ENTRY JSR PC,TYPERR
*RETURN RTS PC
*
*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
*THE ERROR.
*****

```

```

7356 045346 104413          TYPERR: SAVREG
7357 045350 113700 001114          MOVB     $ITEMB,RO      ;ENTER ERROR NUMBER
7358 045354 042700 177400          BIC     #177400,RO      ;CLEAR UNUSED BITS
7359 045360 005300          DEC     RO              ;FORM INDEX FOR ERROR TABLE
7360 045362 006300          ASL     RO
7361 045364 006300          ASL     RO
7362 045366 006300          ASL     RO
7363 045370 062700 001300          1$:     ADD     #$ERRTB,RO  ;FORM ADDRESS OF ERROR ENTRY
7364 045374 012037 045410          MOV     (RO)+,2$        ;GET EM POINTER
7365 045400 001404          BEQ     3$              ;BRANCH IF THERE ISN'T ONE
7366 045402 104401 001211          TYPE    ,SCRLF          ;TYPE CARRIAGE RETURN LINE FEED
7367 045406 104401          TYPE
7368 045410 000000          2$:     .WORD 0          ;TYPE ERROR MESSAGE (EM)
7369 045412 012037 045426          3$:     MOV     (RO)+,4$    ;EM POINTER GOES HERE
7370 045416 001404          BEQ     5$              ;GET DH POINTER
7371 045420 104401 001211          TYPE    ,SCRLF          ;BRANCH IF THERE ISN'T ONE
7372 045424 104401          TYPE
7373 045426 000000          4$:     .WORD 0          ;TYPE CR-LF
7374 045430 012001          5$:     MOV     (RO)+,R1    ;TYPE DATA HEADER
7375 045432 001445          BEQ     20$             ;DH POINTER GOES HERE
                          ;GET DT POINTER
                          ;BRANCH IF THERE ARE NONE

```



7376	045434	005004		CLR	R4		:RESET INDENT SWITCH
7377	045436	012000		MOV	(R0)+,R0		:GET DF POINTER
7378	045440	012002		MOV	(R0)+,R2		:STORE NUMBER OF DH'S
7379	045442	104401	001211	TYPE	,SCRLF		
7380	045446	112003		10\$:	MOVB	(R0)+,R3	:GET & STORE NUMBER OF DATA WORDS
7381	045450	105720		TSTB	(R0)+		:BUMP PAST FORMAT WORD
7382	045452	005703		TST	R3		:TEST IF ANY DATA FOR THIS HEADER
7383	045454	001416		BEQ	14\$		:NO - SKIP DATA PRINT
7384	045456	005704		TST	R4		:CHECK IF INDENT WORDS
7385	045460	001004		BNE	12\$		:YES, GO INDENT
7386	045462	013146		11\$:	MOV	@(R1)+,-(SP)	:PUT FIRST DATA WORD ON STACK
7387	045464	104402		TYPOC			:TYPE IT
7388	045466	005303		DEC	R3		:MORE DATA WORDS
7389	045470	001403		BEQ	13\$		:NO-BRANCH
7390	045472	104401	052267	12\$:	TYPE	,SPACE2	:TYPE SEPARATORS
7391	045476	000771		BR	11\$		:LOOP
7392	045500	104401	001211	13\$:	TYPE	,SCRLF	:TYPE <CR><LF>
7393	045504	005710		TST	(R0)		:CHECK IF NEXT HEADER AVAILBLE
7394	045506	001401		BEQ	14\$		:NO, DO NOT CHANGE INDENT
7395	045510	005104		COM	R4		:CHANGE INDENT
7396	045512	005302		14\$:	DEC	R2	:MORE DH'S?
7397	045514	003414		BLE	20\$		:NO-BRANCH
7398	045516	012037	045536	15\$:	MOV	(R0)+,18\$	:GET NEXT DH POINTER
7399	045522	001751		BEQ	10\$		:IF NO HEADER GET DATA
7400	045524	005704		TST	R4		:INDENT?
7401	045526	001402		BEQ	17\$		:NO-BRANCH
7402	045530	104401	052267	TYPE	,SPACE2		:INDENT
7403	045534	104401		17\$:	TYPE		:TYPE DH
7404	045536	000000		18\$:	.WORD	0	:DH POINTER GOES HERE
7405	045540	104401	001211	TYPE	,SCRLF		
7406	045544	000740		BR	10\$		:LOOP
7407	045546	104414		20\$:	RESREG		
7408	045550	005237	003610	INC	ERRCNT		:INCREMENT ERROR COUNT
7409	045554	032777	010000 133356	BIT	#SW12,@SWR		:CHECK IF ABORT AFTER 20 ERRORS
7410	045562	001421		BEQ	25\$		:NO, RETURN
7411	045564	022737	000024 003610	CMP	#20.,ERRCNT		:CHECK IF EROR THRESHOLD EXCEEDED
7412	045572	103015		BHIS	25\$		:NO, RETURN
7413	045574	104401	052272	TYPE	,ABORT		:TYPE 'PROGRAM HAS BEEN ABORTED BECAUSE
7414							: ERROR THRESHOLD EXCEEDED'
7415	045600	005737	000042	TST	42		:CHECK IF IN CHAIN MODE
7416	045604	001407		BEQ	30\$		:NO, HALT
7417	045606	012737	000001 042406	MOV	#1,\$EOPCT		:FORCE END OF PASS COUNT TO ONE FOR ABORT
7418	045614	012706	001100	MOV	#STACK,SP		:INITIALIZE STACK
7419	045620	000137	042360	JMP	\$EOP		:BRING IN NEXT PROGRAM IN CHAIN
7420	045624	000000		30\$:	HALT		
7421	045626	000207		25\$:	RTS	PC	
7422							
7423				.SBTTL	CONTROLLED PROGRAM HALT		
7424	045630	013702	001270	CTRHLT:	MOV	\$BASE,R2	:SET '611 BASE
7425	045634	012762	100000 000000	MOV	#CCLR,RKCS1(R2)		:CLEAR CONTROLLER
7426	045642	012706	001100	MOV	#STACK,SP		:CLEAR STACK
7427	045646	104401	052223	TYPE	,OPROOF		:TYPE HALT MESSAGE
7428	045652	005737	000042	TST	42		:TEST IF MONITOR PRESENT
7429	045656	001410		BEQ	5\$		:NO - SKIP
7430	045660	105037	001103	CLRB	\$ERFLG		:CLEAR ERROR FLAG
7431	045664	005037	001202	CLR	\$ESCAPE		:CLEAR ESCAPE

```

7432 045670 005037 042406          CLR  SEOPCT          ;SET PASS COUNT TO 0
7433 045674 000137 042360          JMP  SEOP           ;GO TO END OF PASS
7434
7435 045700 000000          5$: HALT           ;HALT PROGRAM
7436 045702 000137 003666          JMP  START1        ;DO RESTART IF CONTINUE
7437
7438          .SBTTL  TYPE ROUTINE
7439
7440          ::*****
7441          ::ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
7442          ::THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
7443          ::NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
7444          ::NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
7445          ::NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
7446          ::
7447          ::CALL:
7448          ::1) USING A TRAP INSTRUCTION
7449          ::          TYPE      ,MESADR          ::MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
7450          ::OR
7451          ::          TYPE
7452          ::          MESADR
7453          ::
7454
7455 045706 105737 001157          $TYPE: TSTB      $STPFLG          ::IS THERE A TERMINAL?
7456 045712 100002          BPL      1$          ::BR IF YES
7457 045714 000000          HALT           ::HALT HERE IF NO TERMINAL
7458 045716 000430          BR      3$          ::LEAVE
7459 045720 010046          1$: MOV      RO,-(SP)          ::SAVE RO
7460 045722 017600 000002          MOV      @2(SP),RO          ::GET ADDRESS OF ASCIZ STRING
7461 045726 122737 000001 001234          CMPB     #APTENV,$ENV          ::RUNNING IN APT MODE
7462 045734 001011          BNE      62$          ::NO,GO CHECK FOR APT CONSOLE
7463 045736 132737 000100 001235          BITB     #APTSPOOL,$ENVM      ::SPOOL MESSAGE TO APT
7464 045744 001405          BEQ      62$          ::NO,GO CHECK FOR CONSOLE
7465 045746 010037 045756          MOV      RO,61$          ::SETUP MESSAGE ADDRESS FOR APT
7466 045752 004737 044706          JSR      PC,$ATY3          ::SPOOL MESSAGE TO APT
7467 045756 000000          .WORD     0          ::MESSAGE ADDRESS
7468 045760 132737 000040 001235          61$: BITB     #APTCSUP,$ENVM      ::APT CONSOLE SUPPRESSED
7469 045766 001003          BNE      60$          ::YES,SKIP TYPE OUT
7470 045770 112046          2$: MOVB     (RO)+,-(SP)          ::PUSH CHARACTER TO BE TYPED ONTO STACK
7471 045772 001005          BNE      4$          ::BR IF IT ISN'T THE TERMINATOR
7472 045774 005726          TST      (SP)+          ::IF TERMINATOR POP IT OFF THE STACK
7473 045776 012600          60$: MOV      (SP)+,RO          ::RESTORE RO
7474 046000 062716 000002          3$: ADD      #2,(SP)          ::ADJUST RETURN PC
7475 046004 000002          RTI           ::RETURN
7476 046006 122716 000011          4$: CMPB     #HT,(SP)          ::BRANCH IF <HT>
7477 046012 001430          BEQ      8$          ::
7478 046014 122716 000200          CMPB     #CRLF,(SP)          ::BRANCH IF NOT <CRLF>
7479 046020 001006          BNE      5$          ::
7480 046022 005726          TST      (SP)+          ::POP <CR><LF> EQUIV
7481 046024 104401          TYPE           ::TYPE A CR AND LF
7482 046026 001211          $CRLF
7483 046030 105037 046164          CLRB     $CHARCNT          ::CLEAR CHARACTER COUNT
7484 046034 000755          BR      2$          ::GET NEXT CHARACTER
7485 046036 004737 046120          5$: JSR      PC,$TYPEC          ::GO TYPE THIS CHARACTER
7486 046042 123726 001156          6$: CMPB     $FILLC,(SP)+          ::IS IT TIME FOR FILLER CHARS.?
7487 046046 001350          BNE      2$          ::IF NO GO GET NEXT CHAR.
    
```

```
7488 046050 013746 001154      MOV      $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
7489                                ;;AND THE NULL CHAR.
7490 046054 105366 000001      7$:  DECB      1(SP)      ;;DOES A NULL NEED TO BE TYPED?
7491 046060 002770                BLT      6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
7492 046062 004737 046120      JSR      PC,$TYPEC      ;;GO TYPE A NULL
7493 046066 105337 046164      DECB      $CHARCNT      ;;DO NOT COUNT AS A COUNT
7494 046072 000770                BR       7$              ;;LOOP
7495
7496                                ;HORIZONTAL TAB PROCESSOR
7497
7498 046074 112716 000040      8$:  MOVB      #' ,(SP)      ;;REPLACE TAB WITH SPACE
7499 046100 004737 046120      9$:  JSR      PC,$TYPEC      ;;TYPE A SPACE
7500 046104 132737 000007 046164  BITB      #7,$CHARCNT      ;;BRANCH IF NOT AT
7501 046112 001372                BNE      9$              ;;TAB STOP
7502 046114 005726                TST      (SP)+          ;;POP SPACE OFF STACK
7503 046116 000724                BR       2$              ;;GET NEXT CHARACTER
7504 046120 105777 133024      $TYPEC: TSTB     @$TPS      ;;WAIT UNTIL PRINTER IS READY
7505 046124 100375                BPL      $TYPEC
7506 046126 116677 000002 133016  MOVB      2(SP),@$TPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
7507 046134 122766 000015 000002  CMPB      #CR,2(SP)      ;;IS CHARACTER A CARRIAGE RETURN?
7508 046142 001003                BNE      1$              ;;BRANCH IF NO
7509 046144 105037 046164      CLRB      $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
7510 046150 000406                BR       $TYPEX          ;;EXIT
7511 046152 122766 000012 000002  1$:  CMPB      #LF,2(SP)      ;;IS CHARACTER A LINE FEED?
7512 046160 001402                BEQ      $TYPEX          ;;BRANCH IF YES
7513 046162 105227                INCB     (PC)+          ;;COUNT THE CHARACTER
7514 046164 000000                $CHARCNT: .WORD 0      ;;CHARACTER COUNT STORAGE
7515 046166 000207                $TYPEX:  RTS      PC
```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```
7516
7517
7518
7519
7520  ;*****
7521  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
7522  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
7523  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
7524  ;*CALL:
7525  ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7526  ;*      TYPOS      ;;CALL FOR TYPEOUT
7527  ;*      .BYTE  N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
7528  ;*      .BYTE  M      ;;M=1 OR 0
7529  ;*                                ;;1=TYPE LEADING ZEROS
7530  ;*                                ;;0=SUPPRESS LEADING ZEROS
7531  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
7532  ;*$TYPOS OR $TYPOC
7533  ;*CALL:
7534  ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7535  ;*      TYPON      ;;CALL FOR TYPEOUT
7536  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
7537  ;*CALL:
7538  ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7539  ;*      TYPOC      ;;CALL FOR TYPEOUT
7540
7541
7542 046170 017646 000000      $TYPOS: MOV      @(SP),-(SP)      ;;PICKUP THE MODE
7543 046174 116637 000001 046413  MOVB      1(SP),$OFILL      ;;LOAD ZERO FILL SWITCH
```

```

7544 046202 112637 046415      MOVB    (SP)+,$SOMODE+1  ;;NUMBER OF DIGITS TO TYPE
7545 046206 062716 000002      ADD     #2,(SP)          ;;ADJUST RETURN ADDRESS
7546 046212 000406                BR      $TYPON
7547 046214 112737 000001 046413 $TYPOC: MOVB    #1,$OFILL      ;;SET THE ZERO FILL SWITCH
7548 046222 112737 000006 046415      MOVB    #6,$SOMODE+1    ;;SET FOR SIX(6) DIGITS
7549 046230 112737 000005 046412 $TYPON: MOVB    #5,$SOCNT  ;;SET THE ITERATION COUNT
7550 046236 010346                MOV     R3,-(SP)        ;;SAVE R3
7551 046240 010446                MOV     R4,-(SP)        ;;SAVE R4
7552 046242 010546                MOV     R5,-(SP)        ;;SAVE R5
7553 046244 113704 046415      MOVB    $SOMODE+1,R4    ;;GET THE NUMBER OF DIGITS TO TYPE
7554 046250 005404                NEG     R4
7555 046252 062704 000006      ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
7556 046256 110437 046414      MOVB    R4,$SOMODE      ;;SAVE IT FOR USE
7557 046262 113704 046413      MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
7558 046266 016605 000012      MOV     12(SP),R5      ;;PICKUP THE INPUT NUMBER
7559 046272 005003                CLR     R3             ;;CLEAR THE OUTPUT WORD
7560 046274 006105 1$:      ROL     R5             ;;ROTATE MSB INTO 'C'
7561 046276 000404                BR      3$
7562 046300 006105 2$:      ROL     R5             ;;GO DO MSB
7563 046302 006105                ROL     R5             ;;FORM THIS DIGIT
7564 046304 006105                ROL     R5
7565 046306 010503                MOV     R5,R3
7566 046310 006103 3$:      ROL     R3             ;;GET LSB OF THIS DIGIT
7567 046312 105337 046414      DECB   $SOMODE          ;;TYPE THIS DIGIT?
7568 046316 100016                BPL     7$             ;;BR IF NO
7569 046320 042703 177770      BIC     #177770,R3     ;;GET RID OF JUNK
7570 046324 001002                BNE     4$             ;;TEST FOR 0
7571 046326 005704                TST     R4             ;;SUPPRESS THIS 0?
7572 046330 001403                BEQ     5$             ;;BR IF YES
7573 046332 005204 4$:      INC     R4             ;;DON'T SUPPRESS ANYMORE 0'S
7574 046334 052703 000060      BIS     #'0,R3         ;;MAKE THIS DIGIT ASCII
7575 046340 052703 000040 5$:      BIS     #' ,R3         ;;MAKE ASCII IF NOT ALREADY
7576 046344 110337 046410      MOVB    R3,8$          ;;SAVE FOR TYPING
7577 046350 104401 046410      TYPE   ,8$            ;;GO TYPE THIS DIGIT
7578 046354 105337 046412 7$:      DECB   $SOCNT          ;;COUNT BY 1
7579 046360 003347                BGT     2$             ;;BR IF MORE TO DO
7580 046362 002402                BLT     6$             ;;BR IF DONE
7581 046364 005204                INC     R4             ;;INSURE LAST DIGIT ISN'T A BLANK
7582 046366 000744                BR      2$             ;;GO DO THE LAST DIGIT
7583 046370 012605 6$:      MOV     (SP)+,R5       ;;RESTORE R5
7584 046372 012604                MOV     (SP)+,R4       ;;RESTORE R4
7585 046374 012603                MOV     (SP)+,R3       ;;RESTORE R3
7586 046376 016666 000002 000004      MOV     2(SP),4(SP)    ;;SET THE STACK FOR RETURNING
7587 046404 012616                MOV     (SP)+,(SP)
7588 046406 000002                RTI
7589 046410 000 8$:      .BYTE  0             ;;RETURN
7590 046411 000                .BYTE  0             ;;STORAGE FOR ASCII DIGIT
7591 046412 000                $SOCNT: .BYTE 0       ;;TERMINATOR FOR TYPE ROUTINE
7592 046413 000                $OFILL: .BYTE 0       ;;OCTAL DIGIT COUNTER
7593 046414 000000      $SOMODE: .WORD 0      ;;ZERO FILL SWITCH
7594                .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
7595
7596                ;;*****
7597                ;;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
7598                ;;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
7599                ;;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED

```

```

7600      ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
7601      ;*REPLACED WITH SPACES.
7602      ;*CALL:
7603      ;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
7604      ;*      TYPDS      ;;GO TO THE ROUTINE
7605
7606      $TYPDS:
7607      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
7608      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
7609      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
7610      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
7611      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
7612      MOV      #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
7613      MOV      20(SP),R5      ;;GET THE INPUT NUMBER
7614      BPL      1$           ;;BR IF INPUT IS POS.
7615      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
7616      MOVB    #'-,1(SP)      ;;MAKE THE ASCII NUMBER NEG.
7617      CLR      R0           ;;ZERO THE CONSTANTS INDEX
7618      MOV      #$DBLK,R3      ;;SETUP THE OUTPUT POINTER
7619      MOVB    #' ,(R3)+      ;;SET THE FIRST CHARACTER TO A BLANK
7620      CLR      R2           ;;CLEAR THE BCD NUMBER
7621      MOV      $DTBL(R0),R1    ;;GET THE CONSTANT
7622      SUB      R1,R5         ;;FORM THIS BCD DIGIT
7623      BLT      4$           ;;BR IF DONE
7624      INC      R2           ;;INCREASE THE BCD DIGIT BY 1
7625      BR      3$
7626      ADD      R1,R5         ;;ADD BACK THE CONSTANT
7627      TST      R2           ;;CHECK IF BCD DIGIT=0
7628      BNE      5$           ;;FALL THROUGH IF 0
7629      TSTB    (SP)          ;;STILL DOING LEADING 0'S?
7630      BMI      7$           ;;BR IF YES
7631      ASLB    (SP)          ;;MSD?
7632      BCC      6$           ;;BR IF NO
7633      MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
7634      BIS      #'0,R2        ;;MAKE THE BCD DIGIT ASCII
7635      BIS      #' ,R2        ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
7636      MOVB    R2,(R3)+      ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
7637      TST      (R0)+        ;;JUST INCREMENTING
7638      CMP      R0,#10        ;;CHECK THE TABLE INDEX
7639      BLT      2$           ;;GO DO THE NEXT DIGIT
7640      BGT      8$           ;;GO TO EXIT
7641      MOV      R5,R2         ;;GET THE LSD
7642      BR      6$           ;;GO CHANGE TO ASCII
7643      TSTB    (SP)+         ;;WAS THE LSD THE FIRST NON-ZERO?
7644      BPL      9$           ;;BR IF NO
7645      MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
7646      CLRB    (R3)          ;;SET THE TERMINATOR
7647      MOV      (SP)+,R5      ;;POP STACK INTO R5
7648      MOV      (SP)+,R3      ;;POP STACK INTO R3
7649      MOV      (SP)+,R2      ;;POP STACK INTO R2
7650      MOV      (SP)+,R1      ;;POP STACK INTO R1
7651      MOV      (SP)+,R0      ;;POP STACK INTO R0
7652      TYPE    $DBLK         ;;NOW TYPE THE NUMBER
7653      MOV      2(SP),4(SP)    ;;ADJUST THE STACK
7654      MOV      (SP)+,(SP)
7655      RTI                    ;;RETURN TO USER

```

7656 046622 023420  
7657 046624 001750  
7658 046626 000144  
7659 046630 000012  
7660 046632 000004  
7661  
7662  
7663  
7664  
7665 046642 000000  
7666 046644 000000  
7667 046646 000000  
7668 046650 000001  
7669 046651  
7670 046652  
7671  
7672  
7673  
7674  
7675  
7676  
7677  
7678  
7679  
7680 046652 005037 046642  
7681 046656 012737 046650 046644  
7682 046664 013737 046644 046646  
7683 046672 012737 046722 000060  
7684 046700 012737 000200 000062  
7685 046706 005777 132234  
7686 046712 012777 000100 132224  
7687 046720 000207  
7688  
7689  
7690  
7691  
7692  
7693  
7694  
7695  
7696 046722 117746 132220  
7697 046726 042716 177600  
7698 046732 021627 000003  
7699 046736 001007  
7700 046740 104401 050036  
7701 046744 004737 046652  
7702 046750 005726  
7703 046752 000137 045630  
7704 046756 021627 000007  
7705 046762 001004  
7706 046764 022737 000176 001140  
7707 046772 001500  
7708  
7709 046774  
7710 046774 022737 000001 046642  
7711 047002 001004

```
SDBLK: .BLKW 4
.SBTTL TTY INPUT ROUTINE

*****
.ENABL LSB
$TKCNT: .WORD 0          ;;NUMBER OF ITEMS IN QUEUE
$TKQIN: .WORD 0          ;;INPUT POINTER
$TKQOUT: .WORD 0         ;;OUTPUT POINTER
$TKQSRV: .BLKB 1        ;;TTY KEYBOARD QUEUE
$TKQEND=.
.EVEN

;*TK INITIALIZE ROUTINE
;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
;*CALL:
;* JSR PC,$TKINT
;* RETURN
$TKINT: CLR $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
        MOV $TKQSRV,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
        MOV $TKQIN,$TKQOUT ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
        MOV $TKSRV,@TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
        MOV #200,@TKVEC+2 ;;'BR' LEVEL 4
        TST @TKB           ;;CLEAR DONE FLAG
        MOV #100,@TKS     ;;ENABLE TTY KEYBOARD INTERRUPT
        RTS PC            ;;RETURN TO CALLER

;*TK SERVICE ROUTINE
;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
;*IT IN THE QUEUE.
;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)
$TKSRV: MOVB @TKB,-(SP)    ;;PICKUP THE CHARACTER
        BIC #^C177,(SP)  ;;STRIP THE JUNK
        CMP (SP),#3      ;;IS IT A CONTROL C?
        BNE 1$           ;;BRANCH IF NO
        TYPE ,SCNTLC     ;;TYPE A CONTROL-C (^C)
        JSR PC,$TKINT    ;;INIT THE KEYBOARD
        TST (SP)+        ;;CLEAN UP STACK
        JMP CTRHLT       ;;CONTROL C RESTART
1$:     CMP (SP),#7      ;;IS IT A CONTROL G?
        BNE 2$           ;;BRANCH IF NO
        CMP #SWREG,SWR   ;;IS SOFT-SWR SELECTED?
        BEQ 6$           ;;GO TO SWR CHANGE
2$:     CMP #1,$TKCNT    ;;IS THE QUEUE FULL?
        BNE 3$           ;;BRANCH IF NO
3$:     ;
```

```
7712 047004 104401 001204          TYPE      ,SBELL      ;;RING THE TTY BELL
7713 047010 005726          TST        (SP)+      ;;CLEAN CHARACTER OFF OF STACK
7714 047012 000451          BR         5$         ;;EXIT
7715 047014 021627 000023      3$:      CMP        (SP),#23   ;;IS IT A CONTROL-S?
7716 047020 001021          BNE       32$        ;;BRANCH IF NO
7717 047022 005077 132116          CLR       @STKS      ;;DISABLE TTY KEYBOARD INTERRUPTS
7718 047026 005726          TST        (SP)+      ;;CLEAN CHAR OFF STACK
7719 047030 105777 132110      31$:     TSTB      @STKS      ;;WAIT FOR A CHAR
7720 047034 100375          BPL       31$        ;;LOOP UNTIL ITS THERE
7721 047036 117746 132104          MOVB     @STKB,-(SP)  ;;GET THE CHARACTER
7722 047042 042716 177600          BIC      #'C177,(SP) ;;MAKE IT 7-BIT ASCII
7723 047046 022627 000021          CMP      (SP)+,#21   ;;IS IT A CONTROL-Q?
7724 047052 001366          BNE       31$        ;;BRANCH IF NO
7725 047054 012777 000100 132062      MOV      #100,@STKS  ;;REENABLE TTY KEYBOARD INTERRUPTS
7726 047062 000002          RTI                    ;;RETURN
7727 047064 005237 046642      32$:     INC      STKCNT     ;;COUNT THIS CHARACTER
7728 047070 021627 000140          CMP      (SP),#140   ;;IS IT UPPER CASE?
7729 047074 002405          BLT      4$         ;;BRANCH IF YES
7730 047076 021627 000175          CMP      (SP),#175   ;;IS IT A SPECIAL CHAR?
7731 047102 003002          BGT      4$         ;;BRANCH IF YES
7732 047104 042716 000040          BIC      #40,(SP)    ;;MAKE IT UPPER CASE
7733 047110 112677 177530      4$:     MOVB     (SP)+,@STKQIN ;;AND PUT IT IN QUEUE
7734 047114 005237 046644          INC      STKQIN      ;;UPDATE THE POINTER
7735 047120 023727 046644 046651      CMP      STKQIN,#STKQEND ;;GO OFF THE END?
7736 047126 001003          BNE       5$        ;;BRANCH IF NO
7737 047130 012737 046650 046644      MOV      #STKQSRST,STKQIN ;;RESET THE POINTER
7738 047136 000002      5$:     RTI                    ;;RETURN
```

```
7739
7740      ;;*****
7741      ;;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
7742      ;;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
7743      ;;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
7744      ;;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
```

```
7745 047140 022737 000176 001140  $CKSWR:  CMP      #SWREG,SWR  ;;IS THE SOFT-SWR SELECTED
7746 047146 001124          BNE      15$        ;;EXIT IF NOT
7747 047150 105777 131770          TSTB     @STKS      ;;IS A CHAR WAITING?
7748 047154 100121          BPL      15$        ;;IF NOT, EXIT
7749 047156 117746 131764          MOVB     @STKB,-(SP) ;;YES
7750 047162 042716 177600          BIC      #'C177,(SP) ;;MAKE IT 7-BIT ASCII
7751 047166 021627 000007          CMP      (SP),#7    ;;IS IT A CONTROL-G?
7752 047172 001300          BNE      2$         ;;IF NOT, PUT IT IN THE TTY QUEUE
7753          ;;AND EXIT
```

```
7754
7755      ;;*****
7756      ;;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
7757      ;;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
7758      ;;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
```

```
7759 047174 123727 001134 000001  6$:     CMPB     $AUTOB,#1   ;;ARE WE RUNNING IN AUTO-MODE?
7760 047202 001674          BEQ      2$         ;;BRANCH IF YES
7761 047204 005726          TST      (SP)+      ;;CLEAR CONTROL-G OFF STACK
7762 047206 004737 046652          JSR     PC,STKINT   ;;FLUSH THE TTY INPUT QUEUE
7763 047212 005077 131726          CLR     @STKS      ;;DISABLE TTY KEYBOARD INTERRUPTS
7764 047216 112737 000001 001135      MOVB     #1,$INTAG  ;;SET INTERRUPT MODE INDICATOR
```

```
7765
7766 047224 104401 050050          TYPE     ,SCNTLG    ;;ECHO THE CONTROL-G (^G)
7767 047230 104401 050055      $GTSWR: TYPE     ,SMSWR  ;;TYPE CURRENT CONTENTS
```

7768	047234	013746	000176		MOV	SWREG,-(SP)	::SAVE SWREG FOR TYPEOUT
7769	047240	104402			TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
7770	047242	104401	050066		TYPE	,SMNEW	::PROMPT FOR NEW SWR
7771	047246	005046		19\$:	CLR	-(SP)	::CLEAR COUNTER
7772	047250	005046			CLR	-(SP)	::THE NEW SWR
7773	047252	105777	131666	7\$:	TSTB	@\$TKS	::CHAR THERE?
7774	047256	100375			BPL	7\$	::IF NOT TRY AGAIN
7775							
7776	047260	117746	131662		MOV	@\$TKB,-(SP)	::PICK UP CHAR
7777	047264	042716	177600		BIC	#^C177,(SP)	::MAKE IT 7-BIT ASCII
7778							
7779	047270	021627	000003		CMP	(SP),#3	::IS IT A CONTROL-C?
7780	047274	001015			BNE	9\$	::BRANCH IF NOT
7781	047276	104401	050036		TYPE	,SCNTLC	::YES, ECHO CONTROL-C (^C)
7782	047302	062706	000006		ADD	#6,SP	::CLEAN UP STACK
7783	047306	123727	001135	000001	CMPB	\$INTAG,#1	::REENABLE TTY KEYBOARD INTERRUPTS?
7784	047314	001003			BNE	8\$	::BRANCH IF NO
7785	047316	012777	000100	131620	MOV	#100,@\$TKS	::ALLOW TTY KEYBOARD INTERRUPTS
7786	047324	000137	045630	8\$:	JMP	CTRHLT	::CONTROL-C RESTART
7787							
7788							
7789	047330	021627	000025	9\$:	CMP	(SP),#25	::IS IT A CONTROL-U?
7790	047334	001005			BNE	10\$	::BRANCH IF NOT
7791	047336	104401	050043		TYPE	,SCNTLU	::YES, ECHO CONTROL-U (^U)
7792	047342	062706	000006	20\$:	ADD	#6,SP	::IGNORE PREVIOUS INPUT
7793	047346	000737			BR	19\$	::LET'S TRY IT AGAIN
7794							
7795							
7796	047350	021627	000015	10\$:	CMP	(SP),#15	::IS IT A <CR>?
7797	047354	001022			BNE	16\$	::BRANCH IF NO
7798	047356	005766	000004		TST	4(SP)	::YES, IS IT THE FIRST CHAR?
7799	047362	001403			BEQ	11\$	::BRANCH IF YES
7800	047364	016677	000002	131546	MOV	2(SP),@SWR	::SAVE NEW SWR
7801	047372	062706	000006	11\$:	ADD	#6,SP	::CLEAN UP STACK
7802	047376	104401	001211	14\$:	TYPE	,SCRLF	::ECHO <CR> AND <LF>
7803	047402	123727	001135	000001	CMPB	\$INTAG,#1	::RE-ENABLE TTY KBD INTERRUPTS?
7804	047410	001003			BNE	15\$	::BRANCH IF NOT
7805	047412	012777	000100	131524	MOV	#100,@\$TKS	::RE-ENABLE TTY KBD INTERRUPTS
7806	047420	000002		15\$:	RTI		::RETURN
7807	047422	004737	046120	16\$:	JSR	PC,\$TYPEC	::ECHO CHAR
7808	047426	021627	000060		CMP	(SP),#60	::CHAR < 0?
7809	047432	002420			BLT	18\$	::BRANCH IF YES
7810	047434	021627	000067		CMP	(SP),#67	::CHAR > 7?
7811	047440	003015			BGT	18\$	::BRANCH IF YES
7812	047442	042726	000060		BIC	#60,(SP)+	::STRIP-OFF ASCII
7813	047446	005766	000002		TST	2(SP)	::IS THIS THE FIRST CHAR
7814	047452	001403			BEQ	17\$	::BRANCH IF YES
7815	047454	006316			ASL	(SP)	::NO, SHIFT PRESENT
7816	047456	006316			ASL	(SP)	::CHAR OVER TO MAKE
7817	047460	006316			ASL	(SP)	::ROOM FOR NEW ONE.
7818	047462	005266	000002	17\$:	INC	2(SP)	::KEEP COUNT OF CHAR
7819	047466	056616	177776		BIS	-2(SP),(SP)	::SET IN NEW CHAR
7820	047472	000667			BR	7\$	::GET THE NEXT ONE
7821	047474	104401	001210	18\$:	TYPE	,\$QUES	::TYPE ?<CR><LF>
7822	047500	000720			BR	20\$	::SIMULATE CONTROL-U.
7823					.DSABL	LSB	



```
7824
7825
7826
7827
7828
7829
7830
7831
7832
7833
7834 047502 011646
7835 047504 016666 000004 000002
7836 047512 005066 000004
7837 047516 005046
7838 047520 012746 047526
7839 047524 000002
7840 047526
7841 047526 005737 046642
7842 047532 001775
7843 047534 005337 046642
7844 047540 117766 177102 000004
7845 047546 005237 046646
7846 047552 023727 046646 046651
7847 047560 001003
7848 047562 012737 046650 046646
7849 047570 000002
7850
7851
7852
7853
7854
7855
7856
7857 047572 010346
7858 047574 005046
7859 047576 012703 050026
7860 047602 022703 050036
7861 047606 101456
7862 047610 104410
7863 047612 112613
7864 047614 122713 000177
7865 047620 001022
7866 047622 005716
7867 047624 001007
7868 047626 112737 000134 050024
7869 047634 104401 050024
7870 047640 012716 177777
7871 047644 005303
7872 047646 020327 050026
7873 047652 103434
7874 047654 111337 050024
7875 047660 104401 050024
7876 047664 000746
7877 047666 005716
7878 047670 001406
7879 047672 112737 000134 050024

*****
*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
*CALL:
*   RDCHR          ;;GET A CHARACTER FROM THE QUEUE
*   RETURN HERE   ;;CHARACTER IS ON THE STACK
*                   ;;WITH PARITY BIT STRIPPED OFF
*
SRDCHR: MOV      (SP),-(SP)      ;;PUSH DOWN THE PC AND
        MOV      4(SP),2(SP)    ;;THE PS
        CLR      4(SP)          ;;GET READY FOR A CHARACTER
        CLR      -(SP)          ;;PUT NEW PS ON STACK
        MOV      #64$,-(SP)     ;;PUT NEW PC ON STACK
        RTI                    ;;POP NEW PC AND PS
64$:
1$:     TST      $STKCNT        ;;WAIT ON A CHARACTER
        BEQ      1$
        DEC      $STKCNT        ;;DECREMENT THE COUNTER
        MOVB    @STKQOUT,4(SP)  ;;GET ONE CHARACTER
        INC      $STKQOUT       ;;UPDATE THE POINTER
        CMP      $STKQOUT,#$STKQEND ;;DID IT GO OFF OF THE END?
        BNE     2$              ;;BRANCH IF NO
        MOV      #$STKQRT,$STKQOUT ;;RESET THE POINTER
        RTI                    ;;RETURN
2$:
*****
*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
*CALL:
*   RDLIN         ;;INPUT A STRING FROM THE TTY
*   RETURN HERE   ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
*                   ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
*
SRDLIN: MOV      R3,-(SP)       ;;SAVE R3
        CLR      -(SP)         ;;CLEAR THE RUBOUT KEY
1$:     MOV      #$TTYIN,R3     ;;GET ADDRESS
2$:     CMP      #$TTYIN+8.,R3  ;;BUFFER FULL?
        BLOS    4$              ;;BR IF YES
        RDCHR   ;;GO READ ONE CHARACTER FROM THE TTY
        MOVB    (SP)+,(R3)     ;;GET CHARACTER
10$:    CMPB    #177,(R3)      ;;IS IT A RUBOUT
        BNE     5$              ;;BR IF NO
        TST     (SP)           ;;IS THIS THE FIRST RUBOUT?
        BNE     6$              ;;BR IF NO
        MOVB    #'\\,9$        ;;TYPE A BACK SLASH
        TYPE    ,9$
6$:     MOV      #-1,(SP)      ;;SET THE RUBOUT KEY
        DEC     R3              ;;BACKUP BY ONE
        CMP     R3,$$TTYIN    ;;STACK EMPTY?
        BLOS    4$              ;;BR IF YES
        MOVB    (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
        MOVB    #'\\,9$        ;;TYPE A BACK SLASH
```

```

7880 047700 104401 050024          TYPE      ,9$
7881 047704 005016                   CLR      (SP)          ;;CLEAR THE RUBOUT KEY
7882 047706 122713 000025      7$:  CMPB   #25,(R3)      ;;IS CHARACTER A CTRL U?
7883 047712 001003                   BNE     8$            ;;BR IF NO
7884 047714 104401 050043          TYPE     ,SCNTLU     ;;TYPE A CONTROL 'U'
7885 047720 000726                   BR      1$            ;;GO START OVER
7886 047722 122713 000022      8$:  CMPB   #22,(R3)      ;;IS CHARACTER A "'R' "?
7887 047726 001011                   BNE     3$            ;;BRANCH IF NO
7888 047730 105013                   CLRB   (R3)          ;;CLEAR THE CHARACTER
7889 047732 104401 001211          TYPE     ,SCRLF     ;;TYPE A "CR" & "LF"
7890 047736 104401 050026          TYPE     ,STTYIN    ;;TYPE THE INPUT STRING
7891 047742 000717                   BR      2$            ;;GO PICKUP ANOTHER CHACTER
7892 047744 104401 001210      4$:  TYPE     ,SQUES     ;;TYPE A '?'
7893 047750 000712                   BR      1$            ;;CLEAR THE BUFFER AND LOOP
7894 047752 111337 050024      3$:  MOVB   (R3),9$      ;;ECHO THE CHARACTER
7895 047756 104401 050024          TYPE     ,9$
7896 047762 122723 000015          CMPB   #15,(R3)+    ;;CHECK FOR RETURN
7897 047766 001305                   BNE     2$            ;;LOOP IF NOT RETURN
7898 047770 105063 177777          CLRB  -1(R3)        ;;CLEAR RETURN (THE 15)
7899 047774 104401 001212          TYPE     ,SLF       ;;TYPE A LINE FEED
7900 050000 005726                   TST   (SP)+          ;;CLEAN RUBOUT KEY FROM THE STACK
7901 050002 012603                   MOV   (SP)+,R3       ;;RESTORE R3
7902 050004 011646                   MOV   (SP),-(SP)     ;;ADJUST THE STACK AND PUT ADDRESS OF THE
7903 050006 016666 000004 000002   MOV   4(SP),2(SP)    ;; FIRST ASCII CHARACTER ON IT
7904 050014 012766 050026 000004   MOV   #$TTYIN,4(SP)
7905 050022 000002                   RTI                    ;;RETURN
7906 050024 000          9$:  .BYTE  0          ;;STORAGE FOR ASCII CHAR. TO TYPE
7907 050025 000          .BYTE  0          ;;TERMINATOR
7908 050026 000010          $TTYIN: .BLKB 8      ;;RESERVE 8 BYTES FOR TTY INPUT
7909 050036 041536 005015 000          $CNTLC: .ASCIZ /^C/<15><12> ;;CONTROL 'C'
7910 050043 0136 006525 000012   $CNTLU: .ASCIZ /^U/<15><12> ;;CONTROL 'U'
7911 050050 043536 005015 000          $CNTLG: .ASCIZ /^G/<15><12> ;;CONTROL 'G'
7912 050055 015 051412 051127   $MSWR:  .ASCIZ <15><12>/SWR = /
7913 050062 036440 000040          $MNEW:  .ASCIZ / NEW = /
7914 050066 020040 042516 020127
7915 050074 020075 000
7916 050100          .EVEN
7917          .SBTTL READ AN OCTAL NUMBER FROM THE TTY
7918
7919          ;*****
7920          ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
7921          ;*CHANGE IT TO BINARY.
7922          ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
7923          ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
7924          ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
7925          ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
7926          ;*CALL:
7927          ;*      RDOCT          ;;READ AN OCTAL NUMBER
7928          ;*      RETURN HERE    ;;LOW ORDER BITS ARE ON TOP OF THE STACK
7929          ;*                    ;;HIGH ORDER BITS ARE IN $HIOCT
7930
7931 050100 011646 000004 000002   $RDOCT: MOV   (SP),-(SP) ;;PROVIDE SPACE FOR THE
7932 050102 016666 000004 000002   MOV   4(SP),2(SP)    ;;INPUT NUMBER
7933 050110 010046                   MOV   R0,-(SP)      ;;PUSH R0 ON STACK
7934 050112 010146                   MOV   R1,-(SP)      ;;PUSH R1 ON STACK
7935 050114 010246                   MOV   R2,-(SP)      ;;PUSH R2 ON STACK

```

```
7936 050116 104411          1$:  RDLIN          ::READ AN ASCIZ LINE
7937 050120 012600          MOV      (SP)+,R0  ::GET ADDRESS OF 1ST CHARACTER
7938 050122 010037 050226  MOV      R0,5$    ::AND SAVE IT
7939 050126 005001          CLR      R1       ::CLEAR DATA WORD
7940 050130 005002          CLR      R2
7941 050132 112046          2$:  MOVB      (R0)+,-(SP) ::PICKUP THIS CHARACTER
7942 050134 001420          SEQ      3$       ::IF ZERO GET OUT
7943 050136 122716 000060  CMPB    #'0,(SP)  ::MAKE SURE THIS CHARACTER
7944 050142 003026          BGT     4$       ::IS AN OCTAL DIGIT
7945 050144 122716 000067  CMPB    #'7,(SP)
7946 050150 002423          BLT     4$
7947 050152 006301          ASL     R1       ::*2
7948 050154 006102          ROL     R2
7949 050156 006301          ASL     R1       ::*4
7950 050160 006102          ROL     R2
7951 050162 006301          ASL     R1       ::*8
7952 050164 006102          ROL     R2
7953 050166 042716 177770  BIC     #'^C7,(SP) ::STRIP THE ASCII JUNK
7954 050172 062601          ADD     (SP)+,R1  ::ADD IN THIS DIGIT
7955 050174 000756          BR      2$       ::LOOP
7956 050176 005726          3$:  TST      (SP)+   ::CLEAN TERMINATOR FROM STACK
7957 050200 010166 000012  MOV     R1,12(SP) ::SAVE THE RESULT
7958 050204 010237 050236  MOV     R2,$HIOCT
7959 050210 012602          MOV     (SP)+,R2  ::POP STACK INTO R2
7960 050212 012601          MOV     (SP)+,R1  ::POP STACK INTO R1
7961 050214 012600          MOV     (SP)+,R0  ::POP STACK INTO R0
7962 050216 000002          RTI                    ::RETURN
7963 050220 005726          4$:  TST      (SP)+   ::CLEAN PARTIAL FROM STACK
7964 050222 105010          CLRB    (R0)      ::SET A TERMINATOR
7965 050224 104401          TYPE                    ::TYPE UP THRU THE BAD CHAR.
7966 050226 000000          5$:  .WORD    0
7967 050230 104401 001210  TYPE    ,SQUES    ::'"?' 'CR' & 'LF'
7968 050234 000730          BR      1$       ::TRY AGAIN
7969 050236 000000          $HIOCT: .WORD    0 ::HIGH ORDER BITS GO HERE
7970          .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
7971
7972          ::*****
7973          ::*SAVE R0-R5
7974          ::*CALL:
7975          ::* SAVREG
7976          ::*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
7977          ::*
7978          ::*TOP---(+16)
7979          ::* +2---(+18)
7980          ::* +4---R5
7981          ::* +6---R4
7982          ::* +8---R3
7983          ::*+10---R2
7984          ::*+12---R1
7985          ::*+14---R0
7986
7987          $SAVREG:
7988          MOV     R0,-(SP)  ::PUSH R0 ON STACK
7989          MOV     R1,-(SP)  ::PUSH R1 ON STACK
7990          MOV     R2,-(SP)  ::PUSH R2 ON STACK
7991          MOV     R3,-(SP)  ::PUSH R3 ON STACK
```

```

7992 050250 010446          MOV     R4,-(SP)      ;;PUSH R4 ON STACK
7993 050252 010546          MOV     R5,-(SP)      ;;PUSH R5 ON STACK
7994 050254 016646 000022    MOV     22(SP),-(SP)  ;;SAVE PS OF MAIN FLOW
7995 050260 016646 000022    MOV     22(SP),-(SP)  ;;SAVE PC OF MAIN FLOW
7996 050264 016646 000022    MOV     22(SP),-(SP)  ;;SAVE PS OF CALL
7997 050270 016646 000022    MOV     22(SP),-(SP)  ;;SAVE PC OF CALL
7998 050274 000002          RTI

8000          ;*RESTORE R0-R5
8001          ;*CALL:
8002          ;*
8003 050276          ;RESREG
8004 050276 012666 000022    MOV     (SP)+,22(SP)  ;;RESTORE PC OF CALL
8005 050302 012666 000022    MOV     (SP)+,22(SP)  ;;RESTORE PS OF CALL
8006 050306 012666 000022    MOV     (SP)+,22(SP)  ;;RESTORE PC OF MAIN FLOW
8007 050312 012666 000022    MOV     (SP)+,22(SP)  ;;RESTORE PS OF MAIN FLOW
8008 050316 012605          MOV     (SP)+,R5      ;;POP STACK INTO R5
8009 050320 012604          MOV     (SP)+,R4      ;;POP STACK INTO R4
8010 050322 012603          MOV     (SP)+,R3      ;;POP STACK INTO R3
8011 050324 012602          MOV     (SP)+,R2      ;;POP STACK INTO R2
8012 050326 012601          MOV     (SP)+,R1      ;;POP STACK INTO R1
8013 050330 012600          MOV     (SP)+,R0      ;;POP STACK INTO R0
8014 050332 000002          RTI

8015
8016          .SBTTL POWER DOWN AND UP ROUTINE
8017
8018          ;;*****
8019
8020          ;POWER DOWN ROUTINE
8021 050334 017737 130600 003634 $PWRDN: MOV     @SWR,SAVSWR  ;SAVE SWITCH REGISTER
8022 050342 012737 050362 000024    MOV     #SPWRUP,PWRVEC ;SET UP VECTOR
8023 050350 012737 000340 000026    MOV     #PR7,PWRVEC+2
8024 050356 000000          HALT
8025 050360 000776          BR     .-2           ;HANG UP
8026
8027          ;;*****
8028
8029          ;POWER UP ROUTINE
8030 050362 005037 050452 050454 $PWRUP: CLR     $PWRCT  ;LOAD WAIT COUNT
8031 050366 012737 000144 050454    MOV     #100,$PWRCT+2
8032 050374 005237 050452          1$: INC     $PWRCT  ;WAIT FOR TELETYPE
8033 050400 001375          BNE     1$
8034 050402 005337 050454          DEC     $PWRCT+2
8035 050406 001372          BNE     1$
8036 050410 012737 050334 000024    MOV     #SPWRDN,PWRVEC ;SET UP FOR POWER DOWN VECTOR
8037 050416 012737 000340 000026    MOV     #PR7,PWRVEC+2
8038 050424 012706 001100          MOV     #STACK,SP     ;FORCE STACK
8039 050430 104401 050456          TYPE   $POWER        ;TYPE POWER
8040 050434 013777 003634 130476    MOV     SAVSWR,@SWR   ;RESTORE SWITCH REGISTER
8041 050442 013702 001270          MOV     $BASE,R2     ;REINITIALISE R2 FOR '611 BASE
8042 050446 000177 130434          JMP     @SLPADR ;GO BACK TO LAST TEST
8043
8044 050452 000000 000000          $PWRCT: .WORD 0,0    ;TELETYPE TIME OUT
8045 050456 047520 042527 000122    $POWER: .ASCIZ /POWER/
8046          .EVEN
8047          .SBTTL TRAP DECODER

```

8048  
8049  
8050  
8051  
8052  
8053  
8054  
8055  
8056  
8057  
8058  
8059  
8060  
8061  
8062  
8063  
8064  
8065  
8066  
8067  
8068  
8069  
8070  
8071  
8072  
8073  
8074  
8075  
8076  
8077  
8078  
8079  
8080  
8081  
8082  
8083  
8084  
8085  
8086  
8087  
8088  
8089  
8090  
8091  
8092

050464 010046  
050466 016600 000002  
050472 005740  
050474 111000  
050476 006300  
050500 016000 050520  
050504 000200

```

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

```

STRAP:  MOV    RO,-(SP)      ;;SAVE R0
        MOV    2(SP),RO     ;;GET TRAP ADDRESS
        TST   -(RO)        ;;BACKUP BY 2
        MOVB  (RO),RO      ;;GET RIGHT BYTE OF TRAP
        ASL   RO           ;;POSITION FOR INDEXING
        MOV   $TRPAD(RO),RO ;;INDEX TO TABLE
        RTS   RO           ;;GO TO ROUTINE
    
```

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

```

STRAP2: MOV    (SP),-(SP)   ;;MOVE THE PC DOWN
        MOV    4(SP),2(SP) ;;MOVE THE PSW DOWN
        RTI                          ;;RESTORE THE PSW
    
```

.SBTTL TRAP TABLE

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

```

:      ROUTINE
:      -----
STRPAD: .WORD  $STRAP2
        $TYPE  ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
        $TYPOC ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOS ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPON ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
        $TYPDS ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

        $GTSWR ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

        $CKSWR ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
        $RDCHR ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
        $RDLIN ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
        $RDOCT ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
        $SAVREG ;;CALL=SAVREG  TRAP+13(104413) SAVE R0-R5 ROUTINE
        $RESREG ;;CALL=RESREG  TRAP+14(104414) RESTORE R0-R5 ROUTINE
        $SCOP1$ ;;CALL=SCOP1$  TRAP+15(104415) INTERNAL LOOP ON ERROR
    
```

```
8093 .SBTTL DATA TABLE FOR PRINT OUT
8094
8095 050554 001220 003604 DT000: .WORD $TESTN,TRAPPC
8096 050560 001220 001116 003500 DT001: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.MR2,T.MR2,E.MR3,T.MR3
8097 050566 003440 003526 003466
8098 050574 003530 003470
8099 050600 001220 001116 003500 DT015: .WORD $TESTN,$ERRPC,E.CS1,T.CS1
8100 050606 003440
8101 050610 001220 001116 003524 DT017: .WORD $TESTN,$ERRPC,E.MR1,T.MR1,PR.BIT,M1.BIT,BITCNT
8102 050616 003464 003614 003616
8103 050624 003622
8104 050626 001220 001116 003500 DT022: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.BA,T.BA,E.WC,T.WC
8105 050634 003440 003504 003444
8106 050642 003502 003442
8107 050646 001220 001116 003524 DT025: .WORD $TESTN,$ERRPC,E.MR1,T.MR1,BITCNT
8108 050654 003464 003622
8109 050660 001220 001116 003500 DT031: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.BA,T.BA,E.WC,T.WC,E.MR1,T.MR1
8110 050666 003440 003504 003444
8111 050674 003502 003442 003524
8112 050702 003464
8113 050704 001220 001116 003500 DT035: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.BA,T.BA
8114 050712 003440 003510 003450
8115 050720 003504 003444
8116 050724 003502 003442
8117 050730 001220 001116 003522 DT041: .WORD E.WC,T.WC
8118 050736 003462 .WORD $TESTN,$ERRPC,E.DB,T.DB
8119 050740 001220 001116 003500 DT042: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2
8120 050746 003440 003510 003450
8121 050754 001220 001116 003522 DT054: .WORD $TESTN,$ERRPC,E.DB,T.DB,WRDCNT
8122 050762 003462 003624
8123 050766 001220 001116 003500 DT055: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,WRDCNT
8124 050774 003440 003510 003450
8125 051002 003624
8126 051004 001220 001116 003500 DT072: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.ER
8127 051012 003440 003510 003450
8128 051020 003514 003454
8129 051024 003504 003444 003502 .WORD E.BA,T.BA,E.WC,T.WC
8130 051032 003442
8131 051034 001220 001116 003500 DT077: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2
8132 051042 003440 003510 003450
8133 051050 001220 001116 003500 DT150: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,WRDCNT,BITCNT
8134 051056 003440 003624 003622
8135 051064 001220 001116 003500 DT164: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS1,BITCNT
8136 051072 003440 003510 003440
8137 051100 003622
8138 051102 001220 001116 003524 DT170: .WORD $TESTN,$ERRPC,E.MR1,T.MR1,P1.BIT,PR.BIT,M1.BIT,M2.BIT,BITCNT,SECCNT
8139 051110 003464 003612 003614
8140 051116 003616 003620 003622
8141 051124 003626
8142 051126 001220 001116 003524 DT171: .WORD $TESTN,$ERRPC,E.MR1,T.MR1
8143 051134 003464
8144 051136 001220 001116 003500 DT175: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.DS,T.DS,E.ER,T.ER
8145 051144 003440 003510 003450
8146 051152 003512 003452 003514
8147 051160 003454
8148 051162 001220 001116 003500 DT211: .WORD $TESTN,$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.DS,T.DS,E.ER,T.ER
```

8149	051170	003440	003510	003450
8150	051176	003512	003452	003514
8151	051204	003454		
8152	051206	003540	003550	003552
8153	051214	003554		

.WORD P.CS1,P.CS2,P.DS,P.ER

```
8154      .SBTTL DATA FORMAT FOR PRINT OUT
8155
8156 051216 000001      DF000: .WORD 1
8157 051220      002    000      .BYTE 2.0
8158 051222 000005      DF001: .WORD 5 ;ERRORS 1-14
8159 051224      000    000      .BYTE 0.0
8160 051226 052403      .WORD DH000A
8161 051230      000    000      .BYTE 0.0
8162 051232 052421      .WORD DH000B
8163 051234      002    000      .BYTE 2.0
8164 051236 052465      .WORD DH001A
8165 051240      000    000      .BYTE 0.0
8166 051242 052544      .WORD DH001B
8167 051244      006    000      .BYTE 6.0
8168 051246 000005      DF015: .WORD 5 ;ERRORS 15-16
8169 051250      000    000      .BYTE 0.0
8170 051252 052403      .WORD DH000A
8171 051254      000    000      .BYTE 0.0
8172 051256 052421      .WORD DH000B
8173 051260      002    000      .BYTE 2.0
8174 051262 052623      .WORD DH015A
8175 051264      000    000      .BYTE 0.0
8176 051266 052642      .WORD DH015B
8177 051270      002    000      .BYTE 2.0
8178 051272 000005      DF017: .WORD 5 ;ERROR 17-21
8179 051274      000    000      .BYTE 0.0
8180 051276 052403      .WORD DH000A
8181 051300      000    000      .BYTE 0.0
8182 051302 052421      .WORD DH000B
8183 051304      002    000      .BYTE 2.0
8184 051306 052660      .WORD DH017A
8185 051310      000    000      .BYTE 0.0
8186 051312 052724      .WORD DH017B
8187 051314      005    000      .BYTE 5.0
8188 051316 000005      DF022: .WORD 5 ;ERROR 22-24
8189 051320      000    000      .BYTE 0.0
8190 051322 052403      .WORD DH000A
8191 051324      000    000      .BYTE 0.0
8192 051326 052421      .WORD DH000B
8193 051330      002    000      .BYTE 2.0
8194 051332 052772      .WORD DH022A
8195 051334      000    000      .BYTE 0.0
8196 051336 053051      .WORD DH022B
8197 051340      006    000      .BYTE 6.0
8198 051342 000005      DF025: .WORD 5 ;ERROR 25
8199 051344      000    000      .BYTE 0.0
8200 051346 052403      .WORD DH000A
8201 051350      000    000      .BYTE 0.0
8202 051352 052421      .WORD DH000B
8203 051354      002    000      .BYTE 2.0
8204 051356 053126      .WORD DH025A
8205 051360      000    000      .BYTE 0.0
8206 051362 053152      .WORD DH025B
8207 051364      003    000      .BYTE 3.0
8208 051366 000005      DF031: .WORD 5 ;ERROR 31-34
8209 051370      000    000      .BYTE 0.0
```



8210	051372	052403		.WORD	DH000A	
8211	051374	000	000	.BYTE	0,0	
8212	051376	052421		.WORD	DH000B	
8213	051400	002	000	.BYTE	2,0	
8214	051402	053200		.WORD	DH031A	
8215	051404	000	000	.BYTE	0,0	
8216	051406	053277		.WORD	DH031B	
8217	051410	010	000	.BYTE	8.,0	
8218	051412	000005		.WORD	5	DF035: ;ERRORS 35-40
8219	051414	000	000	.BYTE	0,0	
8220	051416	052403		.WORD	DH000A	
8221	051420	000	000	.BYTE	0,0	
8222	051422	052421		.WORD	DH000B	
8223	051424	002	000	.BYTE	2,0	
8224	051426	053375		.WORD	DH035A	
8225	051430	000	000	.BYTE	0,0	
8226	051432	053474		.WORD	DH035B	
8227	051434	010	000	.BYTE	8.,0	
8228	051436	000005		.WORD	5	DF041: ;ERROR 41
8229	051440	000	000	.BYTE	0,0	
8230	051442	052403		.WORD	DH000A	
8231	051444	000	000	.BYTE	0,0	
8232	051446	052421		.WORD	DH000B	
8233	051450	002	000	.BYTE	2,0	
8234	051452	053571		.WORD	DH041A	
8235	051454	000	000	.BYTE	0,0	
8236	051456	053606		.WORD	DH041B	
8237	051460	002	000	.BYTE	2,0	
8238	051462	000005		.WORD	5	DF042: ;ERRORS 42-43
8239	051464	000	000	.BYTE	0,0	
8240	051466	052403		.WORD	DH000A	
8241	051470	000	000	.BYTE	0,0	
8242	051472	052421		.WORD	DH000B	
8243	051474	002	000	.BYTE	2,0	
8244	051476	053623		.WORD	DH042A	
8245	051500	000	000	.BYTE	0,0	
8246	051502	053662		.WORD	DH042B	
8247	051504	004	000	.BYTE	4,0	
8248	051506	000005		.WORD	5	DF054: ;ERROR 54
8249	051510	000	000	.BYTE	0,0	
8250	051512	052403		.WORD	DH000A	
8251	051514	000	000	.BYTE	0,0	
8252	051516	052421		.WORD	DH000B	
8253	051520	002	000	.BYTE	2,0	
8254	051522	053720		.WORD	DH054A	
8255	051524	000	000	.BYTE	0,0	
8256	051526	053745		.WORD	DH054B	
8257	051530	003	000	.BYTE	3,0	
8258	051532	000005		.WORD	5	DF055: ;ERROR 55
8259	051534	000	000	.BYTE	0,0	
8260	051536	052403		.WORD	DH000A	
8261	051540	000	000	.BYTE	0,0	
8262	051542	052421		.WORD	DH000B	
8263	051544	002	000	.BYTE	2,0	
8264	051546	053773		.WORD	DH055A	
8265	051550	000	000	.BYTE	0,0	

Line	Key	Value	Code	Field	Format	Value	Field	Format
8266	051552	054040			.WORD	DH055B		
8267	051554	005	000		.BYTE	5,0		
8268	051556	000007		DF072:	.WORD	7		
8269	051560	000	000		.BYTE	0,0		
8270	051562	052403			.WORD	DH000A		
8271	051564	000	000		.BYTE	0,0		
8272	051566	052421			.WORD	DH000B		
8273	051570	002	000		.BYTE	2,0		
8274	051572	054106			.WORD	DH072A		
8275	051574	000	000		.BYTE	0,0		
8276	051576	054145			.WORD	DH072B		
8277	051600	004	000		.BYTE	4,0		
8278	051602	054203			.WORD	DH072C		
8279	051604	000	000		.BYTE	0,0		
8280	051606	054262			.WORD	DH072D		
8281	051610	006	000		.BYTE	6,0		
8282	051612	000007		DF077:	.WORD	7		
8283	051614	000	000		.BYTE	0,0		
8284	051616	052403			.WORD	DH000A		
8285	051620	000	000		.BYTE	0,0		
8286	051622	052421			.WORD	DH000B		
8287	051624	002	000		.BYTE	2,0		
8288	051626	054106			.WORD	DH072A		
8289	051630	000	000		.BYTE	0,0		
8290	051632	054145			.WORD	DH072B		
8291	051634	004	000		.BYTE	4,0		
8292	051636	000005		DF150:	.WORD	5		:ERROR 150
8293	051640	000	000		.BYTE	0,0		
8294	051642	052403			.WORD	DH000A		
8295	051644	000	000		.BYTE	0,0		
8296	051646	052421			.WORD	DH000B		
8297	051650	002	000		.BYTE	2,0		
8298	051652	054337			.WORD	DH150A		
8299	051654	000	000		.BYTE	0,0		
8300	051656	054373			.WORD	DH150B		
8301	051660	004	000		.BYTE	4,0		
8302	051662	000005		DF164:	.WORD	5		:ERROR 164
8303	051664	000	000		.BYTE	0,0		
8304	051666	052403			.WORD	DH000A		
8305	051670	000	000		.BYTE	0,0		
8306	051672	052421			.WORD	DH000B		
8307	051674	002	000		.BYTE	2,0		
8308	051676	054431			.WORD	DH164A		
8309	051700	000	000		.BYTE	0,0		
8310	051702	054475			.WORD	DH164B		
8311	051704	005	000		.BYTE	5,0		
8312	051706	000005		DF170:	.WORD	5		:ERROR 170
8313	051710	000	000		.BYTE	0,0		
8314	051712	052403			.WORD	DH000A		
8315	051714	000	000		.BYTE	0,0		
8316	051716	052421			.WORD	DH000B		
8317	051720	002	000		.BYTE	2,0		
8318	051722	054543			.WORD	DH170A		
8319	051724	000	000		.BYTE	0,0		
8320	051726	054642			.WORD	DH170B		
8321	051730	010	000		.BYTE	8.,0		

8322	051732	000005		DF171:	.WORD	5		:ERRORS 171-174
8323	051734	000	000		.BYTE	0,0		
8324	051736	052403			.WORD	DH000A		
8325	051740	000	000		.BYTE	0,0		
8326	051742	052421			.WORD	DH000B		
8327	051744	002	000		.BYTE	2,0		
8328	051746	054740			.WORD	DH171A		
8329	051750	000	000		.BYTE	0,0		
8330	051752	054757			.WORD	DH171B		
8331	051754	002	000		.BYTE	2,0		
8332	051756	000005		DF175:	.WORD	5		:ERRORS 175-210
8333	051760	000	000		.BYTE	0,0		
8334	051762	052403			.WORD	DH000A		
8335	051764	000	000		.BYTE	0,0		
8336	051766	052421			.WORD	DH000B		
8337	051770	002	000		.BYTE	2,0		
8338	051772	054775			.WORD	DH175A		
8339	051774	000	000		.BYTE	0,0		
8340	051776	055074			.WORD	DH175B		
8341	052000	010	000		.BYTE	8.,0		
8342	052002	000007		DF211:	.WORD	7		:ERRORS 211-214
8343	052004	000	000		.BYTE	0,0		
8344	052006	052403			.WORD	DH000A		
8345	052010	000	000		.BYTE	0,0		
8346	052012	052421			.WORD	DH000B		
8347	052014	002	000		.BYTE	2,0		
8348	052016	054775			.WORD	DH175A		
8349	052020	000	000		.BYTE	0,0		
8350	052022	055074			.WORD	DH175B		
8351	052024	010	000		.BYTE	8.,0		
8352	052026	055171			.WORD	DH211A		
8353	052030	000	000		.BYTE	0,0		
8354	052032	055217			.WORD	DH211B		
8355	052034	004	000		.BYTE	4,0		

```
8356 .SBTTL ASCII MESSAGES
8357
8358 052036 005015 045522 030466 OPR001: .ASCIIZ <15><12>/RK611 VECTOR ADDRESS ( /
8359 052044 020061 042526 052103
8360 052052 051117 040440 042104
8361 052060 042522 051523 024040
8362 052066 000040
8363 052070 024440 036440 000040 OPR002: .ASCIIZ / ) = /
8364 052076 045522 030466 020061 OPR003: .ASCIIZ /RK611 VECTOR ADDRESS ( /
8365 052104 042526 052103 051117
8366 052112 040440 042104 042522
8367 052120 051523 024040 000040
8368 052126 045522 030466 020061 OPR004: .ASCIIZ /RK611 PRIORITY ( /
8369 052134 051120 047511 044522
8370 052142 054524 024040 000040
8371 052150 005015 047062 020104 OPR006: .ASCIIZ <15><12>/2ND PASS RUN TIME IS APPROX 13 MINUTES/<15><12>
8372 052156 040520 051523 051040
8373 052164 047125 052040 046511
8374 052172 020105 051511 040440
8375 052200 050120 047522 020130
8376 052206 031461 046440 047111
8377 052214 052125 051505 005015
8378 052222 000
8379 052223 015 025012 025052 OPR007: .ASCIIZ <15><12>/***** PROGRAM HALTED *****/<15><12>
8380 052230 025052 020040 050040
8381 052236 047522 051107 046501
8382 052244 044040 046101 042524
8383 052252 020104 020040 025052
8384 052260 025052 025052 005015
8385 052266 000
8386 052267 040 000040 SPACE2: .ASCIIZ / /
8387 052272 005015 051120 043517 ABORT: .ASCIIZ <15><12>/PROGRAM ABORTED BECAUSE ERROR THRESHOLD EXCEEDED/<15><12>
8388 052300 040522 020115 041101
8389 052306 051117 042524 020104
8390 052314 042502 040503 051525
8391 052322 020105 051105 047522
8392 052330 020122 044124 042522
8393 052336 044123 046117 020104
8394 052344 054105 042503 042105
8395 052352 042105 005015 000
8396 052357 015 052012 051505 TSTBY1: .ASCIIZ <15><12>/TEST /
8397 052364 020124 000
8398 052367 040 054502 040520 TSTBY2: .ASCIIZ / BYPASSED/<15><12>
8399 052374 051523 042105 005015
8400 052402 000
```

```
8401 .SBTTL DATA HEADERS
8402
8403 052403 124 051505 020124 DH000A: .ASCIZ /TEST ERROR/
8404 052410 020040 042440 051122
8405 052416 051117 000
8406 052421 116 046525 020040 DH000B: .ASCIZ /NUM PC/
8407 052426 020040 050040 000103
8408 052434 042524 052123 020040 DH000C: .ASCII /TEST TRAP/<15><12>
8409 052442 020040 051124 050101
8410 052450 005015
8411 052452 052516 020115 020040 .ASCIZ /NUM PC/
8412 052460 020040 041520 000
8413 052465 105 050130 041505 DH001A: .ASCIZ /EXPECT ACUTAL EXPECT ACTUAL EXPECT ACTUAL/
8414 052472 020124 040440 052503
8415 052500 040524 020114 042440
8416 052506 050130 041505 020124
8417 052514 040440 052103 040525
8418 052522 020114 042440 050130
8419 052530 041505 020124 040440
8420 052536 052103 040525 000114
8421 052544 045522 051503 020061 DH001B: .ASCIZ /RKCS1 RKCS1 MESS A MESS A MESS B MESS B/
8422 052552 020040 045522 051503
8423 052560 020061 020040 042515
8424 052566 051523 040440 020040
8425 052574 042515 051523 040440
8426 052602 020040 042515 051523
8427 052610 041040 020040 042515
8428 052616 051523 041040 000
8429 052623 105 050130 041505 DH015A: .ASCIZ /EXPECT ACTUAL/
8430 052630 020124 040440 052103
8431 052636 040525 000114
8432 052642 045522 051503 020061 DH015B: .ASCIZ /RKCS1 RKCS1/
8433 052650 020040 045522 051503
8434 052656 000061
8435 052660 054105 042520 052103 DH017A: .ASCIZ /EXPECT ACTUAL PRESENT PREVOUS BIT/
8436 052666 020040 041501 052524
8437 052674 046101 020040 051120
8438 052702 051505 047105 020124
8439 052710 051120 053105 052517
8440 052716 020123 044502 000124
8441 052724 045522 051115 020061 DH017B: .ASCIZ /RKMR1 RKMR1 BIT BIT COUNT/
8442 052732 020040 045522 051115
8443 052740 020061 020040 044502
8444 052746 020124 020040 020040
8445 052754 044502 020124 020040
8446 052762 020040 047503 047125
8447 052770 000124
8448 052772 054105 042520 052103 DH022A: .ASCIZ /EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL/
8449 053000 020040 041501 052524
8450 053006 046101 020040 054105
8451 053014 042520 052103 020040
8452 053022 041501 052524 046101
8453 053030 020040 054105 042520
8454 053036 052103 020040 041501
8455 053044 052524 046101 000
8456 053051 122 041513 030523 DH022B: .ASCIZ /RKCS1 RKCS1 RKBA RKBA RKWC RKWC/
```

8457 053056 020040 051040 041513  
8458 053064 030523 020040 051040  
8459 053072 041113 020101 020040  
8460 053100 051040 041113 020101  
8461 053106 020040 051040 053513  
8462 053114 020103 020040 051040  
8463 053122 053513 000103  
8464 053126 054105 042520 052103  
8465 053134 020040 041501 052524  
8466 053142 046101 020040 044502  
8467 053150 000124  
8468 053152 045522 051115 020061  
8469 053160 020040 045522 051115  
8470 053166 020061 020040 047503  
8471 053174 047125 000124  
8472 053200 054105 042520 052103  
8473 053206 020040 041501 052524  
8474 053214 046101 020040 054105  
8475 053222 042520 052103 020040  
8476 053230 041501 052524 046101  
8477 053236 020040 054105 042520  
8478 053244 052103 020040 041501  
8479 053252 052524 046101 020040  
8480 053260 054105 042520 052103  
8481 053266 020040 041501 052524  
8482 053274 046101 000  
8483 053277 122 041513 030523  
8484 053304 020040 051040 041513  
8485 053312 030523 020040 051040  
8486 053320 041113 020101 020040  
8487 053326 051040 041113 020101  
8488 053334 020040 051040 053513  
8489 053342 020103 020040 051040  
8490 053350 053513 020103 020040  
8491 053356 051040 046513 030522  
8492 053364 020040 051040 046513  
8493 053372 030522 000  
8494 053375 105 050130 041505  
8495 053402 020124 040440 052103  
8496 053410 040525 020114 042440  
8497 053416 050130 041505 020124  
8498 053424 040440 052103 040525  
8499 053432 020114 042440 050130  
8500 053440 041505 020124 040440  
8501 053446 052103 040525 114  
8502 053453 040 042440 050130  
8503 053460 041505 020124 040440  
8504 053466 052103 040525 000114  
8505 053474 045522 051503 020061  
8506 053502 020040 045522 051503  
8507 053510 020061 020040 045522  
8508 053516 051503 020062 020040  
8509 053524 045522 051503 020062  
8510 053532 020040 045522 040502  
8511 053540 020040 020040 045522  
8512 053546 040502

DH025A: .ASCIZ /EXPECT ACTUAL BIT/

DH025B: .ASCIZ /RKMR1 RKMR1 COUNT/

DH031A: .ASCIZ /EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL/

DH031B: .ASCIZ /RKCS1 RKCS1 RKBA RKBA RKWC RKWC RKMR1 RKMR1/

DH035A: .ASCII /EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL/

.ASCIZ / EXPECT ACTUAL/

DH035B: .ASCII /RKCS1 RKCS1 RKCS2 RKCS2 RKBA RKBA/

8513	053550	020040	020040	045522		.ASCIZ	/	RKWC	RKWC/						
8514	053556	041527	020040	020040											
8515	053564	045522	041527	000											
8516	053571	105	050130	041505	DH041A:	.ASCIZ	/EXPECT	WORD/							
8517	053576	020124	053440	051117											
8518	053604	000104													
8519	053606	047527	042122	020040	DH041B:	.ASCIZ	/WORD	READ/							
8520	053614	020040	042522	042101											
8521	053622	000													
8522	053623	105	050130	041505	DH042A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL/					
8523	053630	020124	040440	052103											
8524	053636	040525	020114	042440											
8525	053644	050130	041505	020124											
8526	053652	040440	052103	040525											
8527	053660	000114													
8528	053662	045522	051503	020061	DH042B:	.ASCIZ	/RKCS1	RKCS1	RKCS2	RKCS2/					
8529	053670	020040	045522	051503											
8530	053676	020061	020040	045522											
8531	053704	051503	020062	020040											
8532	053712	045522	051503	000062											
8533	053720	054105	042520	052103	DH054A:	.ASCIZ	/EXPECT	WORD	WORD/						
8534	053726	020040	047527	042122											
8535	053734	020040	020040	047527											
8536	053742	042122	000												
8537	053745	127	051117	020104	DH054B:	.ASCIZ	/WORD	READ	COUNT/						
8538	053752	020040	051040	040505											
8539	053760	020104	020040	041440											
8540	053766	052517	052116	000											
8541	053773	105	050130	041505	DH055A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL	WORD/				
8542	054000	020124	040440	052103											
8543	054006	040525	020114	042440											
8544	054014	050130	041505	020124											
8545	054022	040440	052103	040525											
8546	054030	020114	053440	051117											
8547	054036	000104													
8548	054040	045522	051503	020061	DH055B:	.ASCIZ	/RKCS1	RKCS1	RKCS2	RKCS2	COUNT/				
8549	054046	020040	045522	051503											
8550	054054	020061	020040	045522											
8551	054062	051503	020062	020040											
8552	054070	045522	051503	020062											
8553	054076	020040	047503	047125											
8554	054104	000124													
8555	054106	054105	042520	052103	DH072A:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL/					
8556	054114	020040	041501	052524											
8557	054122	046101	020040	054105											
8558	054130	042520	052103	020040											
8559	054136	041501	052524	046101											
8560	054144	000													
8561	054145	122	041513	030523	DH072B:	.ASCIZ	/RKCS1	RKCS1	RKCS2	RKCS2/					
8562	054152	020040	051040	041513											
8563	054160	030523	020040	051040											
8564	054166	041513	031123	020040											
8565	054174	051040	041513	031123											
8566	054202	000													
8567	054203	105	050130	041505	DH072C:	.ASCIZ	/EXPECT	ACTUAL	EXPECT	ACTUAL	EXPECT	ACTUAL/			
8568	054210	020124	040440	052103											

CZ  
CZ







```
8667 .SBTTL ERROR MESSAGES
8668
8669 055254 047125 054105 042520 EM000: .ASCIZ /UNEXPECTED MEMORY PARITY ENABLE TRAP/
8670 055262 052103 042105 046440
8671 055270 046505 051117 020131
8672 055276 040520 044522 054524
8673 055304 042440 040516 046102
8674 055312 020105 051124 050101
8675 055320 000
8676 055321 101 052124 046505 EM200: .ASCIZ /ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER/
8677 055326 052120 047111 020107
8678 055334 047524 041440 042510
8679 055342 045503 051440 042505
8680 055350 020113 042515 051523
8681 055356 043501 020105 051106
8682 055364 046517 051040 040505
8683 055372 020104 042510 042101
8684 055400 051105 000
8685 055403 101 052124 046505 EM201: .ASCIZ /ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER/
8686 055410 052120 047111 020107
8687 055416 047524 041440 042510
8688 055424 045503 051440 042505
8689 055432 020113 042515 051523
8690 055440 043501 020105 051106
8691 055446 046517 053440 044522
8692 055454 042524 044040 040505
8693 055462 042504 000122
8694 055466 052101 042524 050115 EM202: .ASCIZ /ATTEMPTING TO CHECK CLEAR DRIVE MESSAGE FROM READ HEADER/
8695 055474 044524 043516 052040
8696 055502 020117 044103 041505
8697 055510 020113 046103 040505
8698 055516 020122 051104 053111
8699 055524 020105 042515 051523
8700 055532 043501 020105 051106
8701 055540 046517 051040 040505
8702 055546 020104 042510 042101
8703 055554 051105 000
8704 055557 101 052124 046505 EM203: .ASCIZ /ATTEMPTING TO CHECK CLEAR DRIVE MESSAGE FROM WRITE HEADER/
8705 055564 052120 047111 020107
8706 055572 047524 041440 042510
8707 055600 045503 041440 042514
8708 055606 051101 042040 044522
8709 055614 042526 046440 051505
8710 055622 040523 042507 043040
8711 055630 047522 020115 051127
8712 055636 052111 020105 042510
8713 055644 042101 051105 000
8714 055651 101 052124 046505 EM204: .ASCIZ /ATTEMPTING A READ HEADER TO CHECK SECTOR PULSF DETECT/
8715 055656 052120 047111 020107
8716 055664 020101 042522 042101
8717 055672 044040 040505 042504
8718 055700 020122 047524 041440
8719 055706 042510 045503 051440
8720 055714 041505 047524 020122
8721 055722 052520 051514 020105
8722 055730 042504 042524 052103
```

8723	055736	000				
8724	055737	101	052124	046505	EM205: .ASCIZ	/ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT/
8725	055744	052120	047111	020107		
8726	055752	020101	051127	052111		
8727	055760	020105	042510	042101		
8728	055766	051105	052040	020117		
8729	055774	044103	041505	020113		
8730	056002	047111	042504	020130		
8731	056010	052520	051514	020105		
8732	056016	042504	042524	052103		
8733	056024	000				
8734	056025	101	052124	046505	EM206: .ASCIZ	/ATTEMPTING AN NPR READ OF ONE WORD/
8735	056032	052120	047111	020107		
8736	056040	047101	047040	051120		
8737	056046	051040	040505	020104		
8738	056054	043117	047440	042516		
8739	056062	053440	051117	000104		
8740	056070	052101	042524	050115	EM207: .ASCIZ	/ATTEMPTING AN NPR READ/
8741	056076	044524	043516	040440		
8742	056104	020116	050116	020122		
8743	056112	042522	042101	000		
8744	056117	101	052124	046505	EM208: .ASCIZ	/ATTEMPTING NPR READ CHECKING ZERO DETECT/
8745	056124	052120	047111	020107		
8746	056132	050116	020122	042522		
8747	056140	042101	041440	042510		
8748	056146	045503	047111	020107		
8749	056154	042532	047522	042040		
8750	056162	052105	041505	000124		
8751	056170	052101	042524	050115	EM209: .ASCIZ	/ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT INHIBIT/
8752	056176	044524	043516	047040		
8753	056204	051120	051040	040505		
8754	056212	020104	044527	044124		
8755	056220	041040	051525	040440		
8756	056226	042104	042522	051523		
8757	056234	044440	041516	042522		
8758	056242	042515	052116	044440		
8759	056250	044116	041111	052111		
8760	056256	000				
8761	056257	101	052124	046505	EM210: .ASCII	/ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT INHIBIT/
8762	056264	052120	047111	020107		
8763	056272	050116	020122	042522		
8764	056300	042101	053440	052111		
8765	056306	020110	052502	020123		
8766	056314	042101	051104	051505		
8767	056322	020123	047111	051103		
8768	056330	046505	047105	020124		
8769	056336	047111	044510	044502		
8770	056344	124				
8771	056345	040	044103	041505	.ASCIZ	/CHECKING ZERO DETECT/
8772	056352	044513	043516	055040		
8773	056360	051105	020117	042504		
8774	056366	042524	052103	000		
8775	056373	101	052124	046505	EM211: .ASCIZ	/ATTEMPTING TO FORCE NON-EXISTENT MEMORY/
8776	056400	052120	047111	020107		
8777	056406	047524	043040	051117		
8778	056414	042503	047040	047117		

8779	056422	042455	044530	052123	
8780	056430	047105	020124	042515	
8781	056436	047515	054522	000	
8782	056443	101	052124	046505	EM212: .ASCIZ /ATTEMPTING TO CLEAR NON-EXISTENT MEMORY/
8783	056450	052120	047111	020107	
8784	056456	047524	041440	042514	
8785	056464	051101	047040	047117	
8786	056472	042455	044530	052123	
8787	056500	047105	020124	042515	
8788	056506	047515	054522	000	
8789	056513	124	051505	044524	EM213: .ASCIZ /TESTING EXTENDED MEMORY ADDRESSING BITS/
8790	056520	043516	042440	052130	
8791	056526	047105	042504	020104	
8792	056534	042515	047515	054522	
8793	056542	040440	042104	042522	
8794	056550	051523	047111	020107	
8795	056556	044502	051524	000	
8796	056563	101	052124	046505	EM214: .ASCIZ /ATTEMPTING TO FORCE UNIBUS PARITY ERROR/
8797	056570	052120	047111	020107	
8798	056576	047524	043040	051117	
8799	056604	042503	052440	044516	
8800	056612	052502	020123	040520	
8801	056620	044522	054524	042440	
8802	056626	051122	051117	000	
8803	056633	101	052124	046505	EM215: .ASCIZ /ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY/
8804	056640	052120	047111	020107	
8805	056646	050116	020122	042522	
8806	056654	042101	047440	020106	
8807	056662	047514	040503	044524	
8808	056670	047117	050040	044522	
8809	056676	051117	052040	020117	
8810	056704	040502	020104	040520	
8811	056712	044522	054524	000	
8812	056717	101	052124	046505	EM216: .ASCIZ /ATTEMPTING TO CLEAR UNIBUS PARITY ERROR/
8813	056724	052120	047111	020107	
8814	056732	047524	041440	042514	
8815	056740	051101	052440	044516	
8816	056746	052502	020123	040520	
8817	056754	044522	054524	042440	
8818	056762	051122	051117	000	
8819	056767	101	052124	046505	EM217: .ASCIZ /ATTEMPTING 18 BIT NPR READ/
8820	056774	052120	047111	020107	
8821	057002	034061	041040	052111	
8822	057010	047040	051120	051040	
8823	057016	040505	000104		
8824	057022	052101	042524	050115	EM218: .ASCIZ /ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT/
8825	057030	044524	043516	030440	
8826	057036	020070	044502	020124	
8827	057044	050116	020122	042522	
8828	057052	042101	041440	042510	
8829	057060	045503	047111	020107	
8830	057066	042532	047522	042040	
8831	057074	052105	041505	000124	
8832	057102	052101	042524	050115	EM219: .ASCIZ /ATTEMPTING 18 BIT NPR READ WITH BIT 16 (PA) SET/
8833	057110	044524	043516	030440	
8834	057116	020070	044502	020124	

8835 057124 050116 020122 042522  
8836 057132 042101 053440 052111  
8837 057140 020110 044502 020124  
8838 057146 033061 024040 040520  
8839 057154 020051 042523 000124  
8840 057162 052101 042524 050115  
8841 057170 044524 043516 030440  
8842 057176 020070 044502 020124  
8843 057204 050116 020122 042522  
8844 057212 042101 053440 052111  
8845 057220 020110 044502 020124  
8846 057226 033061 024040 040520  
8847 057234 020051 042523 124  
8848 057241 101 052124 046505  
8849 057246 052120 047111 020107  
8850 057254 044523 052515 040514  
8851 057262 044524 047117 047440  
8852 057270 020106 040504 040524  
8853 057276 044440 020116 042522  
8854 057304 042101 044040 040505  
8855 057312 042504 000122  
8856 057316 052101 042524 050115  
8857 057324 044524 043516 051040  
8858 057332 040505 020104 042510  
8859 057340 042101 051105 044440  
8860 057346 020116 040515 047111  
8861 057354 040524 042516 041516  
8862 057362 020105 047515 042504  
8863 057370 000  
8864 057371 101 052124 046505  
8865 057376 052120 047111 020107  
8866 057404 040504 040524 041040  
8867 057412 043125 042506 020122  
8868 057420 042522 042101 040440  
8869 057426 052106 051105 051040  
8870 057434 040505 020104 042510  
8871 057442 042101 051105 000  
8872 057447 101 052124 046505  
8873 057454 052120 047111 020107  
8874 057462 044523 052515 040514  
8875 057470 044524 047117 047440  
8876 057476 020106 040504 040524  
8877 057504 044440 020116 042522  
8878 057512 042101 044040 040505  
8879 057520 042504 020122 030450  
8880 057526 020070 044502 020124  
8881 057534 047506 046522 052101  
8882 057542 000051  
8883 057544 052101 042524 050115  
8884 057552 044524 043516 051040  
8885 057560 040505 020104 042510  
8886 057566 042101 051105 024040  
8887 057574 034061 041040 052111  
8888 057602 043040 051117 040515  
8889 057610 024524 044440 020116  
8890 057616 040515 047111 020124

EM220: .ASCII /ATTEMPTING 18 BIT NPR READ WITH BIT 16 (PA) SET/

EM221: .ASCIZ /ATTEMPTING SIMULATION OF DATA IN READ HEADER/

EM222: .ASCIZ /ATTEMPTING READ HEADER IN MAINTANENCE MODE/

EM223: .ASCIZ /ATTEMPTING DATA BUFFER READ AFTER READ HEADER/

EM224: .ASCIZ /ATTEMPTING SIMULATION OF DATA IN READ HEADER (18 BIT FORMAT)/

EM225: .ASCIZ /ATTEMPTING READ HEADER (18 BIT FORMAT) IN MAINT MODE/

8891	057624	047515	042504	000	
8892	057631	101	052124	046505	EM226: .ASCII /ATTEMPTING DATA BUFFER READ AFTER/<12><15>
8893	057636	052120	047111	020107	
8894	057644	040504	040524	041040	
8895	057652	043125	042506	020122	
8896	057660	042522	042101	040440	
8897	057666	052106	051105	006412	
8898	057674	042522	042101	044040	.ASCIZ /READ HEADER IN 18 BIT FORMAT/
8899	057702	040505	042504	020122	
8900	057710	047111	030440	020070	
8901	057716	044502	020124	047506	
8902	057724	046522	052101	000	
8903	057731	101	052124	046505	EM227: .ASCII /ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER/<15><12>
8904	057736	052120	047111	020107	
8905	057744	047524	041440	042510	
8906	057752	045503	051440	047131	
8907	057760	044103	042040	052105	
8908	057766	041505	020124	047117	
8909	057774	051040	040505	020104	
8910	060002	042510	042101	051105	
8911	060010	005015			
8912	060012	044103	041505	044513	.ASCIZ /CHECKING ZERO DETECT/
8913	060020	043516	055040	051105	
8914	060026	020117	042504	042524	
8915	060034	052103	000		
8916	060037	101	052124	046505	EM230: .ASCII /ATTEMPTING TO CHECK WRITING OF ZEROES BETWEEN INDEX/<15><12>
8917	060044	052120	047111	020107	
8918	060052	047524	041440	042510	
8919	060060	045503	053440	044522	
8920	060066	044524	043516	047440	
8921	060074	020106	042532	047522	
8922	060102	051505	041040	052105	
8923	060110	042527	047105	044440	
8924	060116	042116	054105	005015	
8925	060124	047101	020104	042523	.ASCIZ /AND SECTOR PULSE/
8926	060132	052103	051117	050040	
8927	060140	046125	042523	000	
8928	060145	101	052124	046505	EM231: .ASCII /ATTEMPTING TO RESET WRITE GATE BY SETTING/<15><12>
8929	060152	052120	047111	020107	
8930	060160	047524	051040	051505	
8931	060166	052105	053440	044522	
8932	060174	042524	043440	052101	
8933	060202	020105	054502	051440	
8934	060210	052105	044524	043516	
8935	060216	005015			
8936	060220	042523	052103	051117	.ASCIZ /SECTOR PULSE IN A WRITE HEADER COMMAND/
8937	060226	050040	046125	042523	
8938	060234	044440	020116	020101	
8939	060242	051127	052111	020105	
8940	060250	042510	042101	051105	
8941	060256	041440	046517	040515	
8942	060264	042116	000		
8943	060267	101	052124	046505	EM232: .ASCII /ATTEMPTING TO SET WRITE GATE BY RESETTING/<15><12>
8944	060274	052120	047111	020107	
8945	060302	047524	051440	052105	
8946	060310	053440	044522	042524	

8947	060316	043440	052101	020105	
8948	060324	054502	051040	051505	
8949	060332	052105	044524	043516	
8950	060340	005015			
8951	060342	042523	052103	051117	.ASCIZ /SECTOR PULSE IN A WRITE HEADER COMMAND/
8952	060350	050040	046125	042523	
8953	060356	044440	020116	020101	
8954	060364	051127	052111	020105	
8955	060372	042510	042101	051105	
8956	060400	041440	046517	040515	
8957	060406	042116	000		
8958	060411	101	052124	046505	EM233: .ASCIZ /ATTEMPTING TO WRITE SYNCH OF HEADER/
8959	060416	052120	047111	020107	
8960	060424	047524	053440	044522	
8961	060432	042524	051440	047131	
8962	060440	044103	047440	020106	
8963	060446	042510	042101	051105	
8964	060454	000			
8965	060455	101	052124	046505	EM234: .ASCIZ /ATTEMPTING TO WRITE HEADER DATA/
8966	060462	052120	047111	020107	
8967	060470	047524	053440	044522	
8968	060476	042524	044040	040505	
8969	060504	042504	020122	040504	
8970	060512	040524	000		
8971	060515	101	052124	046505	EM235: .ASCII /ATTEMPTING TO RESET WRITE GATE WITH SECOND/<15><12>
8972	060522	052120	047111	020107	
8973	060530	047524	051040	051505	
8974	060536	052105	053440	044522	
8975	060544	042524	043440	052101	
8976	060552	020105	044527	044124	
8977	060560	051440	041505	047117	
8978	060566	006504	012		
8979	060571	111	042116	054105	.ASCIZ /INDEX PULSE OF WRITE HEADER/
8980	060576	050040	046125	042523	
8981	060604	047440	020106	051127	
8982	060612	052111	020105	042510	
8983	060620	042101	051105	000	
8984	060625	101	052124	046505	EM236: .ASCIZ /ATTEMPTING TO COMPLETE WRITE HEADER IN MAINT MODE/
8985	060632	052120	047111	020107	
8986	060640	047524	041440	046517	
8987	060646	046120	052105	020105	
8988	060654	051127	052111	020105	
8989	060662	042510	042101	051105	
8990	060670	044440	020116	040515	
8991	060676	047111	020124	047515	
8992	060704	042504	000		
8993	060707	101	052124	046505	EM237: .ASCIZ /ATTEMPTING TO WRITE GAP OR DATA SYNCH/
8994	060714	052120	047111	020107	
8995	060722	047524	053440	044522	
8996	060730	042524	043440	050101	
8997	060736	047440	020122	040504	
8998	060744	040524	051440	047131	
8999	060752	044103	000		
9000	060755	101	052124	046505	EM238: .ASCIZ /ATTEMPTING TO WRITE DATA FIELD/
9001	060762	052120	047111	020107	
9002	060770	047524	053440	044522	

9003	060776	042524	042040	052101	
9004	061004	020101	044506	046105	
9005	061012	000104			
9006	061014	052101	042524	050115	EM239: .ASCIZ /ATTEMPTING TO WRITE SYNCH OF HEADER USING 24 SECTOR FORMAT/
9007	061022	047111	020107	047524	
9008	061030	053440	044522	042524	
9009	061036	051440	047131	044103	
9010	061044	047440	020106	042510	
9011	061052	042101	051105	052440	
9012	061060	044523	043516	051040	
9013	061066	020064	042523	052103	
9014	061074	051117	043040	051117	
9015	061102	040515	000124		
9016	061106	052101	042524	050115	EM240: .ASCIZ /ATTEMPTING TO WRITE HEADER DATA USING 24 SECTOR FORMAT/
9017	061114	044524	043516	052040	
9018	061122	020117	051127	052111	
9019	061130	020105	042510	042101	
9020	061136	051105	042040	052101	
9021	061144	020101	051525	047111	
9022	061152	020107	032062	051440	
9023	061160	041505	047524	020122	
9024	061166	047506	046522	052101	
9025	061174	000			
9026	061175	101	052124	046505	EM241: .ASCIZ /ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26 SECTOR)/
9027	061202	052120	047111	020107	
9028	061210	047524	043040	051117	
9029	061216	042503	043040	051117	
9030	061224	040515	020124	051105	
9031	061232	047522	020122	041450	
9032	061240	046506	020124	020075	
9033	061246	033062	051440	041505	
9034	061254	047524	024522	000	
9035	061261	101	052124	046505	EM242: .ASCIZ /ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24 SECTOR)/
9036	061266	052120	047111	020107	
9037	061274	047524	043040	051117	
9038	061302	042503	043040	051117	
9039	061310	040515	020124	051105	
9040	061316	047522	020122	041450	
9041	061324	046506	020124	020075	
9042	061332	032062	051440	041505	
9043	061340	047524	024522	000	
9044	061345	101	052124	046505	EM243: .ASCIZ /ATTEMPTING TO FORCE CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS/
9045	061352	044520	043516	052040	
9046	061360	020117	047506	041522	
9047	061366	020105	047503	052116	
9048	061374	047522	046114	051105	
9049	061402	042440	051122	051117	
9050	061410	053440	052111	020110	
9051	061416	040506	046125	020124	
9052	061424	044502	020124	047111	
9053	061432	042040	044522	042526	
9054	061440	046440	051505	000123	
9055	061446	052101	042524	050115	EM244: .ASCIZ /ATTEMPTING TO CLEAR ERROR/
9056	061454	044524	043516	052040	
9057	061462	020117	046103	040505	
9058	061470	020122	051105	047522	



9059	061476	000122				
9060	061500	047503	046515	047101	EM3000: .ASCIZ	/COMMAND AND STATUS REG 1 INCORRECT/
9061	061506	020104	047101	020104		
9062	061514	052123	052101	051525		
9063	061522	051040	043505	030440		
9064	061530	044440	041516	051117		
9065	061536	042522	052103	000		
9066	061543	115	051505	040523	EM3001: .ASCIZ	/MESSAGE A INCORRECT/
9067	061550	042507	040440	044440		
9068	061556	041516	051117	042522		
9069	061564	052103	000			
9070	061567	115	051505	040523	EM3002: .ASCIZ	/MESSAGE B INCORRECT/
9071	061574	042507	041040	044440		
9072	061602	041516	051117	042522		
9073	061610	052103	000			
9074	061613	103	030523	044440	EM3003: .ASCIZ	/CS1 INCORRECT AFTER SENDING DRIVE CLEAR/
9075	061620	041516	051117	042522		
9076	061626	052103	040440	052106		
9077	061634	051105	051440	047105		
9078	061642	044504	043516	042040		
9079	061650	044522	042526	041440		
9080	061656	042514	051101	000		
9081	061663	103	030523	044440	EM3004: .ASCIZ	/CS1 INCORRECT AFTER AFTER DATA SIMULATION/
9082	061670	041516	051117	042522		
9083	061676	052103	040440	052106		
9084	061704	051105	040440	052106		
9085	061712	051105	042040	052101		
9086	061720	020101	044523	052515		
9087	061726	040514	044524	047117		
9088	061734	000				
9089	061735	115	044501	052116	EM3005: .ASCII	/MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9090	061742	051040	043505	020056		
9091	061750	020061	047111	047503		
9092	061756	051122	041505	020124		
9093	061764	052504	044522	043516		
9094	061772	042040	052101	020101		
9095	062000	044523	052515	040514		
9096	062006	044524	047117			
9097	062012	005015	043101	042524	.ASCIZ	<15><12>/AFTER SECTOR PULSE/
9098	062020	020122	042523	052103		
9099	062026	051117	050040	046125		
9100	062034	042523	000			
9101	062037	115	044501	052116	EM3006: .ASCII	/MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9102	062044	051040	043505	020056		
9103	062052	020061	047111	047503		
9104	062060	051122	041505	020124		
9105	062066	052504	044522	043516		
9106	062074	042040	052101	020101		
9107	062102	044523	052515	040514		
9108	062110	044524	047117			
9109	062114	005015	043101	042524	.ASCIZ	<15><12>/AFTER INDEX PULSE/
9110	062122	020122	047111	042504		
9111	062130	020130	052520	051514		
9112	062136	000105				
9113	062140	040515	047111	020124	EM3007: .ASCII	/MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9114	062146	042522	027107	030440		

9115	062154	044440	041516	051117	
9116	062162	042522	052103	042040	
9117	062170	051125	047111	020107	
9118	062176	040504	040524	051440	
9119	062204	046511	046125	052101	
9120	062212	047511	116		
9121	062215	015	047012	020117	.ASCIZ <15><12>/NO SECTOR OR INDEX PULSE SUPPLIED/
9122	062222	042523	052103	051117	
9123	062230	047440	020122	047111	
9124	062236	042504	020130	052520	
9125	062244	051514	020105	052523	
9126	062252	050120	044514	042105	
9127	062260	000			
9128	062261	102	051525	040440	EM3008: .ASCIZ /BUS ADDRESS INCORRECT AFTER SENDING DRIVE CLEAR/
9129	062266	042104	042522	051523	
9130	062274	044440	041516	051117	
9131	062302	042522	052103	040440	
9132	062310	052106	051105	051440	
9133	062316	047105	044504	043516	
9134	062324	042040	044522	042526	
9135	062332	041440	042514	051101	
9136	062340	000			
9137	062341	127	051117	020104	EM3009: .ASCIZ /WORD COUNT INCORRECT AFTER SENDING DRIVE CLEAR/
9138	062346	047503	047125	020124	
9139	062354	047111	047503	051122	
9140	062362	041505	020124	043101	
9141	062370	042524	020122	042523	
9142	062376	042116	047111	020107	
9143	062404	051104	053111	020105	
9144	062412	046103	040505	000122	
9145	062420	051503	020061	044103	EM3010: .ASCIZ /CS1 CHANGED DURING COMMAND EXECUTION/
9146	062426	047101	042507	020104	
9147	062434	052504	044522	043516	
9148	062442	041440	046517	040515	
9149	062450	042116	042440	042530	
9150	062456	052503	044524	047117	
9151	062464	000			
9152	062465	102	051525	040440	EM3011: .ASCIZ /BUS ADDRESS CHANGED BEFORE INDEX PULSE/
9153	062477	042104	042522	051523	
9154	062500	041440	040510	043516	
9155	062506	042105	041040	043105	
9156	062514	051117	020105	047111	
9157	062522	042504	020130	052520	
9158	062530	051514	000105		
9159	062534	047527	042122	041440	EM3012: .ASCIZ /WORD COUNT CHANGED BEFORE INDEX PULSE/
9160	062542	052517	052116	041440	
9161	062550	040510	043516	042105	
9162	062556	041040	043105	051117	
9163	062564	020105	047111	042504	
9164	062572	020130	052520	051514	
9165	062600	000105			
9166	062602	051503	020061	044103	EM3013: .ASCIZ /CS1 CHANGE AFTER INDEX PULSE/
9167	062610	047101	042507	040440	
9168	062616	052106	051105	044440	
9169	062624	042116	054105	050040	
9170	062632	046125	042523	000	

9171	062637	115	044501	052116	EM3014: .ASCIZ /MAINT REG 1 INCORRECT BEFORE INDEX PULSE/
9172	062644	051040	043505	030440	
9173	062652	044440	041516	051117	
9174	062660	042522	052103	041040	
9175	062666	043105	051117	020105	
9176	062674	047111	042504	020130	
9177	062702	052520	051514	000105	
9178	062710	052502	020123	042101	EM3015: .ASCIZ /BUS ADDRESS CHANGED AFTER INDEX PULSE/
9179	062716	051104	051505	020123	
9180	062724	044103	047101	042507	
9181	062732	020104	043101	042524	
9182	062740	020122	047111	042504	
9183	062746	020130	052520	051514	
9184	062754	000105			
9185	062756	047527	042122	041440	EM3016: .ASCIZ /WORD COUNT CHANGED AFTER INDEX PULSE/
9186	062764	052517	052116	041440	
9187	062772	040510	043516	042105	
9188	063000	040440	052106	051105	
9189	063006	044440	042116	054105	
9190	063014	050040	046125	042523	
9191	063022	000			
9192	063023	103	046517	040515	EM3018: .ASCIZ /COMMAND AND STATUS REG. 2 INCORRECT/
9193	063030	042116	040440	042116	
9194	063036	051440	040524	052524	
9195	063044	020123	042522	027107	
9196	063052	031040	044440	041516	
9197	063060	051117	042522	052103	
9198	063066	000			
9199	063067	102	051525	040440	EM3019: .ASCIZ /BUS ADD REG INCORRECT/
9200	063074	042104	051040	043505	
9201	063102	044440	041516	051117	
9202	063110	042522	052103	000	
9203	063115	127	051117	020104	EM3020: .ASCIZ /WORD COUNT REG INCORRECT/
9204	063122	047503	047125	020124	
9205	063130	042522	020107	047111	
9206	063136	047503	051122	041505	
9207	063144	000124			
9208	063146	040504	040524	051040	EM3021: .ASCIZ /DATA READ INCORRECT/
9209	063154	040505	020104	047111	
9210	063162	047503	051122	041505	
9211	063170	000124			
9212	063172	051503	020061	047111	EM3022: .ASCIZ /CS1 INCORRECT AFTER READING DATA BUFFER/
9213	063200	047503	051122	041505	
9214	063206	020124	043101	042524	
9215	063214	020122	042522	042101	
9216	063222	047111	020107	040504	
9217	063230	040524	041040	043125	
9218	063236	042506	000122		
9219	063242	051503	020062	047111	EM3023: .ASCIZ /CS2 INCORRECT AFTER READING DATA BUFFER/
9220	063250	047503	051122	041505	
9221	063256	020124	043101	042524	
9222	063264	020122	042522	042101	
9223	063272	047111	020107	040504	
9224	063300	040524	041040	043125	
9225	063306	042506	000122		
9226	063312	051105	047522	020122	EM3024: .ASCIZ /ERROR REG INCORRECT/

9227	063320	042522	020107	047111	
9228	063326	047503	051122	041505	
9229	063334	000124			
9230	063336	040515	047111	020124	EM3025: .ASCIZ /MAINT REG 1 INCORRECT AFTER INDEX PULSE/
9231	063344	042522	020107	020061	
9232	063352	047111	047503	051122	
9233	063360	041505	020124	043101	
9234	063366	042524	020122	047111	
9235	063374	042504	020130	052520	
9236	063402	051514	000105		
9237	063406	051503	020061	047111	EM3026: .ASCIZ /CS1 INCORRECT AFTER COMMAND COMPLETION/
9238	063414	047503	051122	041505	
9239	063422	020124	043101	042524	
9240	063430	020122	047503	046515	
9241	063436	047101	020104	047503	
9242	063444	050115	042514	044524	
9243	063452	047117	000		
9244	063455	103	031123	044440	EM3027: .ASCIZ /CS2 INCORRECT AFTER COMMAND COMPLETION/
9245	063462	041516	051117	042522	
9246	063470	052103	040440	052106	
9247	063476	051105	041440	046517	
9248	063504	040515	042116	041440	
9249	063512	046517	046120	052105	
9250	063520	047511	000116		
9251	063524	051503	020061	047111	EM3028: .ASCIZ /CS1 INCORRECT AFTER READING EMPTY SILO/
9252	063532	047503	051122	041505	
9253	063540	020124	043101	042524	
9254	063546	020122	042522	042101	
9255	063554	047111	020107	046505	
9256	063562	052120	020131	044523	
9257	063570	047514	000		
9258	063573	103	031123	044440	EM3029: .ASCIZ /CS2 INCORRECT AFTER READING EMPTY SILO/
9259	063600	041516	051117	042522	
9260	063606	052103	040440	052106	
9261	063614	051105	051040	040505	
9262	063622	044504	043516	042440	
9263	063630	050115	054524	051440	
9264	063636	046111	000117		
9265	063642	040515	047111	020124	EM3030: .ASCIZ /MAINT REG 1 INCORRECT/
9266	063650	042522	020107	020061	
9267	063656	047111	047503	051122	
9268	063664	041505	000124		
9269	063670	051104	053111	020105	EM3031: .ASCIZ /DRIVE STATUS REG INCORRECT/ !
9270	063676	052123	052101	051525	
9271	063704	051040	043505	044440	
9272	063712	041516	051117	042522	
9273	063720	052103	000		
9274	063723	105	051122	051117	EM3032: .ASCIZ /ERROR REG INCORRECT/
9275	063730	051040	043505	044440	
9276	063736	041516	051117	042522	
9277	063744	052103	000		
9278	063747	115	030522	044440	EMW1: .ASCIZ /MR1 INCORRECT ON 1ST UPWARD TRANSITION OF MAINT CLOCK/
9279	063754	041516	051117	042522	
9280	063762	052103	047440	020116	
9281	063770	051461	020124	050125	
9282	063776	040527	042122	052040	

9283	064004	040522	051516	051511
9284	064012	044524	047117	047440
9285	064020	020106	040515	047111
9286	064026	020124	046103	041517
9287	064034	000113		
9288	064036	051115	020061	047111
9289	064044	047503	051122	041505
9290	064052	020124	047117	030440
9291	064060	052123	042040	053517
9292	064066	053516	051101	020104
9293	064074	051124	047101	044523
9294	064102	052123	047511	020116
9295	064110	043117	046440	044501
9296	064116	052116	041440	047514
9297	064124	045503	000	
9298	064127	115	030522	044440
9299	064134	041516	051117	042522
9300	064142	052103	047440	020116
9301	064150	047062	020104	050125
9302	064156	040527	042122	052040
9303	064164	040522	051516	051511
9304	064172	044524	047117	047440
9305	064200	020106	040515	047111
9306	064206	020124	046103	041517
9307	064214	000113		
9308	064216	051115	020061	047111
9309	064224	047503	051122	041505
9310	064232	020124	047117	031040
9311	064240	042116	042040	053517
9312	064246	053516	051101	020104
9313	064254	051124	047101	044523
9314	064262	052123	047511	020116
9315	064270	043117	046440	044501
9316	064276	052116	041440	047514
9317	064304	045503	000	

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

EMW3: .ASCIZ /MR1 INCORRECT ON 2ND UPWARD TRANSITION OF MAINT CLOCK/

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

9318			
9319			
9320		064310	
9321	064310	000000	
9322	064312	177777	
9323	064314	125252	
9324	064316	052515	
9325	064320	101706	
9326	064322	060422	
9327	064324	130715	
9328	064326	177777	
9329	064330	000000	
9330	064332	177777	
9331	064334	000000	
9332	064336	177777	
9333	064340	000000	
9334	064342	125252	
9335	064344	052525	
9336	064346	125252	
9337	064350	052525	
9338	064352	125252	
9339	064354	052525	
9340	064356	044444	
9341	064360	022222	
9342	064362	111111	
9343	064364	052012	
9344	064366	100520	
9345	064370	052012	
9346	064372	155555	
9347	064374	066666	
9348	064376	155555	
9349	064400	104210	
9350	064402	104210	
9351	064404	104210	
9352	064406	100	100
9353	064410	101	101
9354	064412	102	102
9355	064414	103	103
9356	064416	104	104
9357	064420	105	105
9358	064422	106	106
9359	064424	107	107
9360	064426	110	110
9361	064430	111	111
9362	064432	112	112
9363	064434	113	113
9364	064436	114	114
9365	064440	115	115
9366	064442	116	116
9367	064444	117	117
9368	064446	120	120
9369	064450	121	121
9370	064452	122	122
9371	064454	123	123
9372	064456	124	124
9373	064460	125	125

.SBTTL DATA PATTERNS

.EVEN	
PATTERN:	.WORD 000000
	.WORD 177777
	.WORD 125252
	.WORD 052515
	.WORD 101706
	.WORD 060422
	.WORD 130715
HEAD1:	.WORD 177777
	.WORD 000000
	.WORD 177777
HEAD2:	.WORD 000000
	.WORD 177777
	.WORD 000000
HEAD3:	.WORD 125252
	.WORD 052525
	.WORD 125252
	.WORD 052525
	.WORD 125252
HEAD4:	.WORD 052525
	.WORD 044444
	.WORD 022222
	.WORD 111111
HEAD5:	.WORD 052012
	.WORD 100520
	.WORD 052012
HEAD6:	.WORD 155555
	.WORD 066666
	.WORD 155555
HEAD7:	.WORD 104210
	.WORD 104210
	.WORD 104210
NPRBUF:	.BYTE 100,100
	.BYTE 101,101
	.BYTE 102,102
	.BYTE 103,103
	.BYTE 104,104
	.BYTE 105,105
	.BYTE 106,106
	.BYTE 107,107
	.BYTE 110,110
	.BYTE 111,111
	.BYTE 112,112
	.BYTE 113,113
	.BYTE 114,114
	.BYTE 115,115
	.BYTE 116,116
	.BYTE 117,117
	.BYTE 120,120
	.BYTE 121,121
	.BYTE 122,122
	.BYTE 123,123
	.BYTE 124,124
	.BYTE 125,125

9374	064462	126	126	.BYTE	126,126
9375	064464	127	127	.BYTE	127,127
9376	064466	130	130	.BYTE	130,130
9377	064470	131	131	.BYTE	131,131
9378	064472	132	132	.BYTE	132,132
9379	064474	133	133	.BYTE	133,133
9380	064476	134	134	.BYTE	134,134
9381	064500	135	135	.BYTE	135,135
9382	064502	136	136	.BYTE	136,136
9383	064504	137	137	.BYTE	137,137
9384	064506	140	140	.BYTE	140,140
9385	064510	141	141	.BYTE	141,141
9386	064512	142	142	.BYTE	142,142
9387	064514	143	143	.BYTE	143,143
9388	064516	144	144	.BYTE	144,144
9389	064520	145	145	.BYTE	145,145
9390	064522	146	146	.BYTE	146,146
9391	064524	147	147	.BYTE	147,147
9392	064526	150	150	.BYTE	150,150
9393	064530	151	151	.BYTE	151,151
9394	064532	152	152	.BYTE	152,152
9395	064534	153	153	.BYTE	153,153
9396	064536	154	154	.BYTE	154,154
9397	064540	155	155	.BYTE	155,155
9398	064542	156	156	.BYTE	156,156
9399	064544	157	157	.BYTE	157,157
9400	064546	160	160	.BYTE	160,160
9401	064550	161	161	.BYTE	161,161
9402	064552	162	162	.BYTE	162,162
9403	064554	163	163	.BYTE	163,163
9404	064556	164	164	.BYTE	164,164
9405	064560	165	165	.BYTE	165,165
9406	064562	166	166	.BYTE	166,166
9407	064564	167	167	.BYTE	167,167
9408	064566	170	170	.BYTE	170,170
9409	064570	171	171	.BYTE	171,171
9410	064572	172	172	.BYTE	172,172
9411	064574	173	173	.BYTE	173,173
9412	064576	174	174	.BYTE	174,174
9413	064600	175	175	.BYTE	175,175
9414	064602	176	176	.BYTE	176,176
9415	064604	177	177	.BYTE	177,177
9416	064606	200	200	.BYTE	200,200
9417	064610	201	201	.BYTE	201,201
9418	064612	202	202	.BYTE	202,202
9419	064614	203	203	.BYTE	203,203
9420	064616	204	204	.BYTE	204,204
9421	064620	205	205	.BYTE	205,205
9422	064622	206	206	.BYTE	206,206
9423	064624	207	207	.BYTE	207,207
9424	064626	210	210	.BYTE	210,210
9425	064630	000102		WRBUFF: .BLKW	66.
9426		064644		BADPAR= WRBUFF+14	
9427		000001		.END	

ABASE = 177440	1008#	1246	1287				
ABORT = 052272	7413	8387#					
ACDW1 = 000000	1246	1289					
ACDW2 = 000000	1246	1290					
ACLO = 000010	1103#						
ACPUOP = 000000	1246	1261					
ADDW0 = 000000	1246						
ADDW1 = 000000	1246						
ADDW10 = 000000	1246						
ADDW11 = 000000	1246						
ADDW12 = 000000	1246						
ADDW13 = 000000	1246						
ADDW14 = 000000	1246						
ADDW15 = 000000	1246						
ADDW2 = 000000	1246						
ADDW3 = 000000	1246						
ADDW4 = 000000	1246						
ADDW5 = 000000	1246						
ADDW6 = 000000	1246						
ADDW7 = 000000	1246						
ADDW8 = 000000	1246						
ADDW9 = 000000	1246						
ADEVCT = 000000	1246	1252					
ADEVN = 000000	1246	1288					
AENV = 000000	1246	1257					
AENVN = 000000	1246	1258					
AFATAL = 000000	1246	1249					
AMADR1 = 000000	1246	1274					
AMADR2 = 000000	1246	1278					
AMADR3 = 000000	1246	1281					
AMADR4 = 000000	1246	1284					
AMAMS1 = 000000	1246	1268					
AMAMS2 = 000000	1246	1276					
AMAMS3 = 000000	1246	1279					
AMAMS4 = 000000	1246	1282					
AMSGAD = 000000	1246	1254					
AMSGLG = 000000	1246	1255					
AMSGTY = 000000	1246	1248					
AMTYP1 = 000000	1246	1269					
AMTYP2 = 000000	1246	1277					
AMTYP3 = 000000	1246	1280					
AMTYP4 = 000000	1246	1283					
APASS = 000000	1246	1251					
APRIOR = 000005	1007#	1246					
APTCSU = 000040	7290#	7468					
APTENV = 000001	7246	7288#	7322	7461			
APTSIZ = 000200	2281	7287#					
APTSPO = 000100	7248	7289#	7463				
ASWREG = 000000	1246	1259					
ATESTN = 000000	1246	1250					
AUNIT = 000000	1246	1253					
AUSWR = 000000	1246	1260					
AVECT1 = 120210	1006#	1246	1285				
AVECT2 = 000000	1246	1286					
BADPAR = 064644	9426#						
BAI = 000020	1066#	3508	3523	3553	3582	3595	













EM3022	063172	1509	1575	1641	1845	1915	1950	1986	9212#					
EM3023	063242	1515	1581	1647	1851	1921	1956	1992	9219#					
EM3024	063312	1665	1773	9226#										
EM3025	063336	1473	9230#											
EM3026	063406	1938	1974	9237#										
EM3027	063455	1944	1980	9244#										
EM3028	063524	2016	9251#											
EM3029	063573	2022	9258#											
EM3030	063642	2032	2037	2042	9265#									
EM3031	063670	2065	2089	2113	2137	9269#								
EM3032	063723	2071	2095	2119	2143	9274#								
ERRCNT	003610	2211#	2285*	7408*	7411									
ERRVEC=	000004	955#	2266	2267*	2278*	7052	7053	7057*	7062*	7074*	7079*	7088*	7098*	7099*
		7130	7131*	7133*	7136*									
E.ASOF	003516	2174#												
E.BA	003504	2169#	2827*	2834	2867	2892	2924	2955	2986*	3034	3086*	3113	3119*	3127
		3150	3201*	3228	3240	3246*	3254*	3270	3330*	3357	3369	3375*	3384*	3403
		3433*	3446	3457*	3504*	3544	3572	3622*	3665	3753*	3761	3786*	3824	3830*
		3916*	3924	3949*	3987	3993*	4079*	4087	4112*	4150	4156*	4192*	4218	4233
		4239*	4248*	4260	8104	8109	8113	8129						
E.CS1	003500	2167#	2400*	2403	2445*	2448	2488*	2491	2532*	2535	2576*	2577	2634*	2635
		2665*	2666	2715*	2716	2746*	2747	2789*	2790	2826*	2829	2862	2887	2919
		2952	2987*	3028	3048	3078*	3121	3144	3169	3198*	3234	3264	3295	3327*
		3363	3397	3420	3440	3476	3501*	3538	3566	3596	3626*	3656	3674*	3676
		3726*	3755	3788*	3818	3889*	3918	3951*	3981	4052*	4081	4114*	4144	4193*
		4227	4254	4283	4338*	4350	4372*	4374	4389	4448*	4460	4482*	4484	4499
		4558*	4570	4592*	4594	4609	4668*	4680	4702*	4704	4719	4778*	4790	4812*
		4814	4829	4887*	4899	4921*	4923	4938	4982*	4988	5003*	5005	5056*	5068
		5090*	5092	5107	5310*	5311	5430*	5431	5549*	5550	5668*	5669	5787*	5788
		5906*	5907	6025*	6026	6155*	6156	6312*	6313	6458*	6459	6576*	6577	6616*
		6620	6642*	6646	6693*	6697	6719*	6723	6772*	6776	6798*	6802	8096	8099
		8104	8109	8113	8119	8123	8126	8131	8133	8135	8144	8148		
E.CS2	003510	2171#	3027*	3031	3047*	3051	3104*	3124	3147	3168*	3172	3219*	3237	3250*
		3267	3279*	3294*	3298	3348*	3366	3380*	3400	3409*	3423	3435*	3443	3459*
		3475*	3479	3523*	3541	3553*	3569	3582*	3595*	3599	3652*	3655*	3659	3675*
		3679	3752*	3758	3817*	3821	3915*	3921	3980*	3984	4078*	4084	4143*	4147
		4208*	4226*	4230	4257	4267*	4282*	4286	4373*	4377	4388*	4394	4483*	4487
		4498*	4504	4593*	4597	4608*	4614	4703*	4707	4718*	4724	4813*	4817	4828*
		4834	4922*	4926	4937*	4943	4983*	4993	5004*	5008	5091*	5095	5106*	5112
		6617*	6623	6643*	6649	6694*	6700	6720*	6726	6773*	6779	6799*	6805	8113
		8119	8123	8126	8131	8135	8144	8148						
E.DA	003506	2170#												
E.DB	003522	2176#	3041*	3042	3160*	3161	3281*	3282	3411*	3412	3462*	3463	3579*	3583
		3768*	3769	3837*	3838	3931*	3932	4000*	4001	4094*	4095	4163*	4164	4271*
		4272	4382*	4399	4492*	4509	4602*	4619	4712*	4729	4822*	4839	4931*	4948
		5100*	5117	8117	8121									
E.DCYL	003520	2175#												
E.DS	003512	2172#	6618*	6626	6644*	6652	6695*	6703	6721*	6729	6774*	6782	6800*	6808
		8144	8148											
E.ECPS	003532	2180#												
E.ECPT	003534	2181#												
E.ER	003514	2173#	3651*	3662	6619*	6629	6645*	6655	6696*	6706	6722*	6732	6775*	6785
		6801*	6811	8126	8144	8148								
E.MR1	003524	2177#	2585*	2592	2601*	2605	2617	2627	2680*	2686	2698	2709	2754*	2760
		2772	2783	2844*	2852	2909	2951*	2961	5161*	5181*	5182*	5185	5188*	5189*
		5192	5244*	5300*	5301	5364*	5420*	5421	5483*	5539*	5540	5602*	5658*	5659

		5721*	5777*	5778	5840*	5896*	5897	5959*	6015*	6016	6087*	6145*	6146	6212*
		6275*	6281*	6301*	6302	6358*	6421*	6427*	6447*	6448	6508*	6566*	6567	6869*
		6872	6883*	6884*	6889*	6894*	6899*	6902	6913*	6916	6927*	6928*	6931	6948*
		6951*	6954*	6957	6966*	6969	6978*	6981	8101	8107	8109	8138	8142	
E.MR2	003526	2178#	2401*	2408	2446*	2453	2489*	2496	2533*	2540	8096			
E.MR3	003530	2179#	2402*	2411	2447*	2456	2490*	2499	2534*	2543	8096			
E.SPAC	003536	2182#												
E.WC	003502	2168#	2828*	2839	2872	2897	2929	2958	2988*	3037	3087*	3088*	3089	3120*
		3130	3133	3153	3203*	3243	3247*	3253*	3273	3332*	3372	3376*	3377	3383*
		3406	3434*	3449	3458*	3506*	3532	3547	3550*	3556*	3575	3624*	3668	3754*
		3764	3784*	3811	3827	3831*	3917*	3927	3947*	3974	3990	3994*	4080*	4090
		4110*	4137	4153	4157*	4190*	4224	4236	4240*	4249*	4263	8104	8109	8116
		8129												
FMT	= 000020	1085#	6619	6696										
GNS	= ***** U	1147	2305	6839	6846	8078	8079	8080	8081	8082	8084	8086	8087	8088
		8089	8090	8091	8092									
GO	= 000001	1048#	3626	4372	4482	4592	4702	4812	4921	5003	5090	5310	5430	5549
		5668	5787	5906	6025	6155	6312	6458	6576	6616	6693	6772		
GTSWR	= 104406	2300	8084#											
HEAD1	064326	4337	4381	4886	4930	5055	5099	5219	5262	6060	6061	6481	6482	9328#
HEAD2	064334	4447	4491	5339	5382	9331#								
HEAD3	064342	4557	4601	5458	5501	6185	6186	9334#						
HEAD4	064356	4667	4711	5577	5620	9340#								
HEAD5	064364	4777	4821	5696	5739	9343#								
HEAD6	064372	5815	5858	9346#										
HEAD7	064400	5934	5977	9349#										
HT	= 000011	865#	7476	7517										
HVRC	= 000400	1089#												
IDAE	= 002000	1091#												
IE	= 000100	1049#												
ILF	= 000001	1081#												
INTR	= 000300	1044#												
IOTVEC	= 000020	960#	2250*	2251*										
IR	= 000100	1068#	3027	3047	3104	3168	3219	3279	3294	3348	3409	3459	3475	3523
		3582	3595	3652	3675	3752	3817	3915	3980	4078	4143	4208	4267	4282
		4373	4388	4483	4498	4593	4608	4703	4718	4813	4828	4922	4937	4983
		5004	5091	5106	6617	6643	6694	6720	6773	6799				
KIPAR0	= 172340	993#	7038	7066										
KIPAR1	= 172342	994#												
KIPAR2	= 172344	995#												
KIPAR3	= 172346	996#												
KIPAR4	= 172350	997#												
KIPAR5	= 172352	998#												
KIPAR6	= 172354	999#	3715*	3773*	3878*	3936*	4041*	4099*						
KIPAR7	= 172356	1000#	7082											
KIPDR0	= 172300	982#												
KIPDR1	= 172302	983#												
KIPDR2	= 172304	984#												
KIPDR3	= 172306	985#												
KIPDR4	= 172310	986#												
KIPDR5	= 172312	987#												
KIPDR6	= 172314	988#												
KIPDR7	= 172316	989#												
LF	= 000012	866#	7511	7517										
MCLK	= 000400	1122#	2393	2438	2481	2525	2571	2660	2674	2741	2819	2847	2849	2879
		2904	2906	2936	2943	3004	3010	3016	3093	3099	3106	3136	3208	3214







CZR6CDO RK611 DSKLS CTRL PRT3  
CZR6CD.P11 19-JUN-80 13:49

MACY11 30A(1052) 19-JUN-80 13:52 PAGE 187  
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0185

P.WC	003542	2187#																		
P1.BIT	003612	2212#	5169*	5246*	5257*	5268	5271*	5274*	5285	5286*	5366*	5377*	5388	5391*						
		5394*	5405	5406*	5485*	5496*	5507	5510*	5513*	5524	5525*	5604*	5615*	5626						
		5629*	5632*	5643	5644*	5723*	5734*	5745	5748*	5751*	5762	5763*	5842*	5853*						
		5864	5867*	5870*	5881	5882*	5961*	5972*	5983	5986*	5989*	6000	6001*	6089*						
		6100*	6110	6113*	6116*	6127	6128*	6214*	6225*	6235	6238*	6241*	6253	6254*						
		6259*	6267	6268*	6360*	6371*	6381	6384*	6387*	6399	6400*	6405*	6413	6414*						
		6510*	6521*	6531	6534*	6537*	6548	6549*	6890	6944	8138									
RDBIT	043540	2590	2603	2615	2625	2684	2696	2707	2758	2770	2781	4331	4335	4348						
		4367	4441	4445	4458	4477	4551	4555	4568	4587	4661	4665	4678	4697						
		4771	4775	4788	4807	4880	4884	4897	4916	4985	5049	5053	5066	5085						
		7000#																		
RDCMR =	104410	7862	8087#																	
RDDATA=	000021	1039#																		
RDGATE=	100000	1129#	2601																	
RDHEAD=	000025	1041#	2390	2400	2478	2488	2569	2576	2634	2658	2665	2715	2739	2746						
		2789	4320	4338	4372	4430	4448	4482	4540	4558	4592	4650	4668	4702						
		4760	4778	4812	4869	4887	4921	4972	4982	5003	5036	5056	5090	6600						
		6616	6677	6693																
RDLIN =	104411	7936	8088#																	
RDOCT =	104412	2316	2327	2344	8089#															
RDY =	000200	1050#	3626	3674	4372	4482	4592	4702	4812	4921	5003	5090	5310	5430						
		5549	5668	5787	5906	6025	6155	6312	6458	6576	6616	6642	6693	6719						
		6772	6798																	
RECAL =	000013	1036#																		
RESREG=	104414	7407	8091#																	
RESTR	003652	1152	2230#																	
RESVEC=	000010	956#																		
RKASOF=	000016	1019#																		
RKBA =	000004	1014#	2815*	2824	2860	2885	2917	2949	2998*	3021	3085*	3111	3142	3202*						
		3226	3262	3331*	3355	3395	3438	3505*	3530	3564	3623*	3648	3746	3787*						
		3809	3909	3950*	3972	4072	4113*	4135	4191*	4216	4252	5145*	5219*	5339*						
		5458*	5577*	5696*	5815*	5934*	6060*	6185*	6331*	6481*	6755*									
RKCS1 =	000000	1012#	2385*	2390*	2397	2406*	2430*	2435*	2442	2451*	2473*	2478*	2485	2494*						
		2517*	2522*	2529	2538*	2565*	2569*	2575	2580*	2595*	2608*	2620*	2630*	2633						
		2654*	2655*	2658*	2664	2669*	2689*	2701*	2711*	2714	2735*	2736*	2739*	2745						
		2750*	2763*	2775*	2785*	2788	2811*	2817*	2823	2859	2884	2916	2948	2964*						
		2993*	3001*	3020	3045	3083*	3090*	3110	3140	3164	3199*	3205*	3225	3260						
		3285	3328*	3334*	3354	3393	3415	3436	3466	3502*	3509*	3529	3562	3586						
		3620*	3627*	3646	3671*	3672	3723*	3727*	3745	3782*	3789*	3808	3844*	3886*						
		3890*	3908	3945*	3952*	3971	4007*	4049*	4053*	4071	4108*	4115*	4134	4170*						
		4187*	4194*	4215	4250	4278	4318*	4320*	4349	4357*	4370	4384	4428*	4430*						
		4459	4467*	4480	4494	4538*	4540*	4569	4577*	4590	4604	4648*	4650*	4679						
		4687*	4700	4714	4758*	4760*	4789	4797*	4810	4824	4867*	4869*	4898	4906*						
		4919	4933	4970*	4972*	4986	5001	5011*	5034*	5036*	5067	5075*	5088	5102						
		5141*	5147*	5217*	5221*	5309	5337*	5341*	5429	5456*	5460*	5548	5575*	5579*						
		5667	5694*	5698*	5786	5813*	5817*	5905	5932*	5936*	6024	6058*	6063*	6154						
		6183*	6188*	6311	6329*	6334*	6457	6479*	6484*	6575	6600*	6608	6612	6637*						
		6638	6677*	6685	6689	6714*	6715	6756*	6764	6768	6793*	6794	7425*							
RKCS2 =	000010	1016#	2389*	2434*	2477*	2521*	2999*	3026	3046	3118	3141	3165	3233	3261						
		3291	3362	3394	3419	3437	3472	3508*	3537	3563	3592	3647	3673	3751						
		3816	3914	3979	4077	4142	4223	4251	4279	4371	4385	4481	4495	4591						
		4605	4701	4715	4811	4825	4920	4934	4987	5002	5089	5103	6597*	6613						
		6639	6674*	6690	6716	6751*	6769	6795												
RKDA =	000006	1015#	2388*	2433*	2476*	2520*	2997*													
RKDB =	000024	1021#	3040	3159	3280	3410	3461	3581	3767	3836	3930	3999	4093	4162						

RKDCYL= 000020  
RKDS = 000012  
RKECPS= 000030  
RKECPT= 000032  
RKER = 000014  
RKMR1 = 000026

4270	4383	4493	4603	4713	4823	4932	5000	5101						
1020#	2387*	2432*	2475*	2519*	2996*	6599*	6676*	6753*						
1017#	6614	6640	6691	6717	6770	6796								
1025#														
1026#														
1018#	3650	6615	6641	6692	6718	6771	6797							
1022#	2386*	2393*	2394*	2431*	2438*	2439*	2474*	2481*	2482*	2518*	2525*	2526*		
2566*	2567*	2568*	2571*	2572*	2583*	2584*	2591	2604	2616	2626	2656*	2657*		
2660*	2661*	2672*	2674*	2675*	2678*	2685	2697	2708	2737*	2738*	2741*	2742*		
2759	2771	2782	2812*	2813*	2814*	2819*	2820*	2847*	2848*	2849*	2850*	2851		
2878*	2879*	2880*	2883*	2904*	2905*	2906*	2907*	2908	2934*	2936*	2937*	2940*		
2943*	2944*	2947	2995*	3004*	3005*	3009*	3010*	3011*	3014*	3016*	3017*	3084*		
3093*	3094*	3098*	3099*	3100*	3103*	3106*	3107*	3136*	3137*	3200*	3208*	3209*		
3213*	3214*	3215*	3218*	3221*	3222*	3256*	3257*	3329*	3337*	3338*	3342*	3343*		
3344*	3347*	3350*	3351*	3389*	3390*	3427*	3428*	3503*	3512*	3513*	3517*	3518*		
3519*	3522*	3525*	3526*	3558*	3559*	3621*	3630*	3631*	3635*	3636*	3637*	3640*		
3642*	3643*	3724*	3729*	3730*	3734*	3735*	3736*	3739*	3741*	3742*	3783*	3791*		
3792*	3796*	3797*	3798*	3801*	3804*	3805*	3887*	3892*	3893*	3897*	3898*	3899*		
3902*	3904*	3905*	3946*	3954*	3955*	3959*	3960*	3961*	3964*	3967*	3968*	4050*		
4055*	4056*	4060*	4061*	4062*	4065*	4067*	4068*	4109*	4117*	4118*	4122*	4123*		
4124*	4127*	4130*	4131*	4188*	4197*	4198*	4202*	4203*	4204*	4207*	4211*	4212*		
4244*	4245*	4319*	4322*	4323*	4326*	4327*	4429*	4432*	4433*	4436*	4437*	4539*		
4542*	4543*	4546*	4547*	4649*	4652*	4653*	4656*	4657*	4759*	4762*	4763*	4766*		
4767*	4868*	4871*	4872*	4875*	4876*	4971*	4974*	4975*	4978*	4979*	5035*	5038*		
5039*	5042*	5043*	5044*	5045*	5142*	5143*	5144*	5150*	5151*	5155*	5156*	5157*		
5160*	5164*	5165*	5183*	5184	5190*	5191	5218*	5224*	5225*	5229*	5230*	5231*		
5234*	5236*	5237*	5240*	5241*	5292*	5294*	5295*	5298*	5299	5305*	5306*	5338*		
5344*	5345*	5349*	5350*	5351*	5354*	5356*	5357*	5360*	5361*	5412*	5414*	5415*		
5418*	5419	5425*	5426*	5457*	5463*	5464*	5468*	5469*	5470*	5473*	5475*	5476*		
5479*	5480*	5531*	5533*	5534*	5537*	5538	5544*	5545*	5576*	5582*	5583*	5587*		
5588*	5589*	5592*	5594*	5595*	5598*	5599*	5650*	5652*	5653*	5656*	5657	5663*		
5664*	5695*	5701*	5702*	5706*	5707*	5708*	5711*	5713*	5714*	5717*	5718*	5769*		
5771*	5772*	5775*	5776	5782*	5783*	5814*	5820*	5821*	5825*	5826*	5827*	5830*		
5832*	5833*	5836*	5837*	5888*	5890*	5891*	5894*	5895	5901*	5902*	5933*	5939*		
5940*	5944*	5945*	5946*	5949*	5951*	5952*	5955*	5956*	6007*	6009*	6010*	6013*		
6014	6020*	6021*	6059*	6066*	6067*	6071*	6072*	6073*	6077*	6080*	6081*	6084*		
6085*	6137*	6139*	6140*	6143*	6144	6150*	6151*	6184*	6191*	6192*	6196*	6197*		
6198*	6201*	6203*	6204*	6209*	6210*	6293*	6295*	6296*	6299*	6300	6307*	6308*		
6330*	6337*	6338*	6342*	6343*	6344*	6347*	6349*	6350*	6355*	6356*	6439*	6441*		
6442*	6445*	6446	6453*	6454*	6480*	6487*	6488*	6492*	6493*	6494*	6498*	6501*		
6502*	6505*	6506*	6558*	6560*	6561*	6564*	6565	6571*	6572*	6598*	6602*	6603*		
6606*	6675*	6679*	6680*	6683*	6752*	6758*	6759*	6762*	6870*	6871	6900*	6901		
6914*	6915	6929*	6930	6955*	6956	6967*	6968	6979*	6980	7004*	7007*	7008*		
7009*	7010*	7013*	7014*	7015*	7016*									
1023#	2398	2443	2486	2530										
1024#	2399	2444	2487	2531										
2205#	2360*													
1027#														
2204#	2358*	2359*												
1013#	2816*	2825	2861	2886	2918	2950	3000*	3022	3089*	3112	3143	3204*		
3227	3263	3333*	3356	3396	3439	3507*	3531	3565	3625*	3649	3725*	3747		
3785*	3810	3888*	3910	3948*	3973	4051*	4073	4111*	4136	4189*	4217	4253		
5146*	5220*	5340*	5459*	5578*	5697*	5816*	5935*	6062*	6187*	6333*	6483*	6754*		
1065#														
7356	8090#													
2221#	8021*	8040												

RKMR2 = 000034  
RKMR3 = 000036  
RKPRI 003602  
RKSPAR= 000022  
RKVEC 003600  
RKWC = 000002

RLS = 000010  
SAVREG= 104413  
SAVSWR 003634



S.RECL=	000040	1134#																		
S.RTC =	000200	1136#	2446																	
S.SEEK=	000020	1133#	2401	2446																
S.STSP=	000100	1135#																		
S.UNLD=	002000	1139#																		
TBITVE=	000014	957#																		
TKVEC =	000060	964#	7683*	7684*																
TPVEC =	000064	965#																		
TRAPPC	003604	2206#	7022*	8095																
TRAPVE=	000034	963#	2254*	2255*																
TRTVEC=	000014	958#																		
TSTBY1	052357	3708	3871	4034	8396#															
TSTBY2	052367	3711	3874	4037	8398#															
TST1	004650	2366	2382#	7184																
TST10	007432	2751	2764	2776	2786	2791	2808#	7191												
TST11	010556	2981#	7192																	
TST12	011244	3057	3073#	7193																
TST13	012130	3181	3195#	7194																
TST14	013016	3324#	7195																	
TST15	014222	3498#	7196																	
TST16	015100	3617#	7197																	
TST17	015536	3680	3698#	7198																
TST2	005066	2407	2412	2427#	7185															
TST20	016652	3712	3861#	7199																
TST21	017766	3875	4024#	7200																
TST22	021102	4038	4184#	7201																
TST23	021772	4315#	7202																	
TST24	022524	4358	4392	4397	4425#	7203														
TST25	023256	4468	4502	4507	4535#	7204														
TST26	024010	4578	4612	4617	4645#	7205														
TST27	024542	4688	4722	4727	4755#	7206														
TST3	005304	2452	2457	2470#	7186															
TST30	025274	4798	4832	4837	4864#	7207														
TST31	026026	4907	4941	4946	4967#	7208														
TST32	026326	4991	4996	5031#	7209															
TST33	027074	5076	5110	5115	5138#	7210														
TST34	027464	5193	5214#	7211																
TST35	030310	5312	5334#	7212																
TST36	031134	5432	5453#	7213																
TST37	031760	5551	5572#	7214																
TST4	005520	2495	2500	2514#	7187															
TST40	032604	5670	5691#	7215																
TST41	033430	5789	5810#	7216																
TST42	034254	5908	5929#	7217																
TST43	035100	6027	6055#	7218																
TST44	035740	6157	6180#	7219																
TST45	036764	6314	6326#	7220																
TST46	040010	6460	6476#	7221																
TST47	040650	6578	6594#	7222																
TST5	005734	2539	2544	2562#	7188															
TST50	041274	6671#	7223																	
TST51	041720	6748#	7224																	
TST6	006416	2581	2596	2609	2621	2631	2636	2651#	7189											
TST7	007044	2670	2690	2702	2712	2717	2732#	7190												
TYPDS =	104405	6843	6850	8082#																
TYPE =	104401	2292	2309	2312	2315	2322	2326	2334	2343	3708	3711	3871	3874	4034						

		4037	6837	6844	6851	7312	7320	7366	7367	7371	7372	7379	7390	7392
		7402	7403	7405	7413	7427	7481	7577	7652	7700	7712	7766	7767	7770
		7781	7791	7802	7821	7869	7875	7880	7884	7889	7890	7892	7895	7899
		7965	7967	8039	8078#									
TYPERR	045346	7319	7356#											
TYPOC =	104402	2314	2325	2342	3710	3873	4036	7387	7769	8079#				
TYPON =	104404	8081#												
TYPOS =	104403	8080#												
T.ASOF	003456	2155#												
T.BA	003444	2150#	2824*	2834	2860*	2867	2885*	2892	2917*	2924	2949*	2955	3021*	3034
		3111*	3127	3142*	3150	3226*	3240	3262*	3270	3355*	3369	3395*	3403	3438*
		3446	3530*	3544	3564*	3572	3648*	3665	3746*	3761	3809*	3824	3909*	3924
		3972*	3987	4072*	4087	4135*	4150	4216*	4233	4252*	4260	8104	8109	8113
		8129												
T.CS1	003440	2148#	2397*	2403	2442*	2448	2485*	2491	2529*	2535	2575*	2577	2633*	2635
		2664*	2666	2714*	2716	2745*	2747	2788*	2790	2823*	2829	2859*	2862	2884*
		2887	2916*	2919	2948*	2952	3020*	3028	3045*	3048	3110*	3121	3140*	3144
		3164*	3169	3225*	3234	3260*	3264	3285*	3295	3354*	3363	3393*	3397	3415*
		3420	3436*	3440	3466*	3476	3529*	3538	3562*	3566	3586*	3596	3646*	3656
		3672*	3676	3745*	3755	3808*	3818	3908*	3918	3971*	3981	4071*	4081	4134*
		4144	4215*	4227	4250*	4254	4278*	4283	4349*	4350	4370*	4374	4384*	4389
		4459*	4460	4480*	4484	4494*	4499	4569*	4570	4590*	4594	4604*	4609	4679*
		4680	4700*	4704	4714*	4719	4789*	4790	4810*	4814	4824*	4829	4898*	4899
		4919*	4923	4933*	4938	4986*	4988	5001*	5005	5067*	5068	5088*	5092	5102*
		5107	5309*	5311	5429*	5431	5548*	5550	5667*	5669	5786*	5788	5905*	5907
		6024*	6026	6154*	6156	6311*	6313	6457*	6459	6575*	6577	6612*	6620	6632
		6638*	6646	6689*	6697	6709	6715*	6723	6768*	6776	6788	6794*	6802	8096
		8099	8104	8109	8113	8119	8123	8126	8131	8133	8135	8144	8148	
T.CS2	003450	2152#	3026*	3031	3046*	3051	3118*	3124	3141*	3147	3165*	3172	3233*	3237
		3261*	3267	3291*	3298	3362*	3366	3394*	3400	3419*	3423	3437*	3443	3472*
		3479	3537*	3541	3563*	3569	3592*	3599	3647*	3653	3659	3673*	3679	3751*
		3758	3816*	3821	3914*	3921	3979*	3984	4077*	4084	4142*	4147	4223*	4230
		4251*	4257	4279*	4286	4371*	4377	4385*	4394	4481*	4487	4495*	4504	4591*
		4597	4605*	4614	4701*	4707	4715*	4724	4811*	4817	4825*	4834	4920*	4926
		4934*	4943	4987*	4993	5002*	5008	5089*	5095	5103*	5112	6613*	6623	6633
		6639*	6649	6690*	6700	6710	6716*	6726	6769*	6779	6789	6795*	6805	8113
		8119	8123	8126	8131	8144	8148							
T.DA	003446	2151#												
T.DB	003462	2157#	3040*	3042	3159*	3161	3280*	3282	3410*	3412	3461*	3463	3581*	3583
		3767*	3769	3836*	3838	3930*	3932	3999*	4001	4093*	4095	4162*	4164	4270*
		4272	4383*	4399	4493*	4509	4603*	4619	4713*	4729	4823*	4839	4932*	4948
		5101*	5117	8117	8121									
T.DCYL	003460	2156#												
T.DS	003452	2153#	6614*	6626	6634	6640*	6652	6691*	6703	6711	6717*	6729	6770*	6782
		6790	6796*	6808	8144	8148								
T.ECPS	003472	2161#												
T.ECPT	003474	2162#												
T.ER	003454	2154#	3650*	3662	6615*	6629	6635	6641*	6655	6692*	6706	6712	6718*	6732
		6771*	6785	6791	6797*	6811	8126	8144	8148					
T.MR1	003464	2158#	2591*	2592	2604*	2605	2616*	2617	2626*	2627	2685*	2686	2697*	2698
		2708*	2709	2759*	2760	2771*	2772	2782*	2783	2851*	2852	2908*	2909	2947*
		2961	5184*	5185	5191*	5192	5299*	5301	5419*	5421	5538*	5540	5657*	5659
		5776*	5778	5895*	5897	6014*	6016	6144*	6146	6300*	6302	6446*	6448	6565*
		6567	6871*	6872	6901*	6902	6915*	6916	6930*	6931	6956*	6957	6968*	6969
		6980*	6981	8101	8107	8109	8138	8142						
T.MR2	003466	2159#	2398*	2408	2443*	2453	2486*	2496	2530*	2540	8096			

T.MR3	003470	2160#	2399*	2411	2444*	2456	2487*	2499	2531*	2543	8096			
T.SPAR	003476	2163#												
T.WC	003442	2149#	2825*	2839	2861*	2872	2886*	2897	2918*	2929	2950*	2958	3022*	3037
		3112*	3130	3143*	3153	3227*	3243	3263*	3273	3356*	3372	3396*	3406	3439*
		3449	3531*	3547	3565*	3575	3649*	3668	3747*	3764	3810*	3827	3910*	3927
		3973*	3990	4073*	4090	4136*	4153	4217*	4236	4253*	4263	8104	8109	8116
		8129												
UFE	= 000400	1070#												
UNLOAD	= 000007	1034#												
UNS	= 040000	1095#												
UPE	= 020000	1075#												
VV	= 000100	1106#												
WAITIM	003632	2220#	6607	6684	6763									
WCE	= 040000	1076#												
WLE	= 004000	1092#												
WRBUFF	064630	2815	2827	2986	2994*	2998	5145	6755	9425#	9426				
WRDATA	= 000023	1040#												
WRDCNT	003624	2217#	3158*	3175*	3278*	3286	3301*	3386*	3452*	3467	3482*	3580*	3587	3602*
		4269*	4289*	4352*	4353*	4380*	4386	4402*	4403	4462*	4463*	4490*	4496	4512*
		4513	4572*	4573*	4600*	4606	4622*	4623	4682*	4683*	4710*	4716	4732*	4733
		4792*	4793*	4820*	4826	4842*	4843	4901*	4902*	4929*	4935	4951*	4952	5070*
		5071*	5098*	5104	5120*	5121	8121	8123	8133					
WRHEAD	= 000027	1042#	2435	2445	2522	2532	2817	2826	2987	3001	3078	3090	3198	3205
		3327	3334	3501	3509	3626	3627	3726	3727	3788	3789	3889	3890	3951
		3952	4052	4053	4114	4115	4193	4194	5147	5221	5310	5341	5430	5460
		5549	5579	5668	5698	5787	5817	5906	5936	6025	6063	6155	6188	6312
		6334	6458	6484	6576	6756	6772							
WRL	= 004000	1109#												
WRTBIT	042570	5176	5252	5258	5275	5287	5372	5378	5395	5407	5491	5497	5514	5526
		5610	5616	5633	5645	5729	5735	5752	5764	5848	5854	5871	5883	5967
		5973	5990	6002	6095	6101	6117	6129	6220	6226	6242	6255	6260	6269
		6276	6283	6366	6372	6388	6401	6406	6415	6422	6429	6516	6522	6538
		6550	6869#											
WRTCHK	= 000031	1043#												
WRTGAT	= 040000	1128#	2951	5161	5181	5189	5244	5364	5483	5602	5721	5840	5959	6087
		6212	6358	6508	6884									
WR.PAR	= 000004	1004#												
SAPTHD	001000	1178	1184#											
SASTAT	= ***** U	7268	7283											
SATYC	044724	7239	7241#											
SATY1	044700	7237#												
SATY3	044706	7238#	7466											
SATY4	044716	7240#	7325											
SAUTOB	001134	1215#	2302*	7759	7916									
SBASE	001270	1287#	2313	2321*	2362	2384	2429	2472	2516	2564	2653	2734	2810	2983
		3075	3197	3326	3500	3619	3714	3877	4040	4186	4317	4427	4537	4647
		4757	4866	4969	5033	5140	5216	5336	5455	5574	5693	5812	5931	6057
		6182	6328	6478	6596	6673	6750	7424	8041					
SBDADR	001122	1210#												
SBDAT	001126	1212#												
SBELL	001204	1238#	7312	7345	7712	7909								
SCDW1	001274	1289#												
SCDW2	001276	1290#												
SCHARC	046164	7483*	7493*	7500	7509*	7514#								
SCKSWR	047140	7745#	8086											
SCMTAG	001100	1198#	2243	2244	2252	2258	2259	2260						















CZR6CDO RK611 DSKLS CTRL PRT3  
CZR6CD.P11 19-JUN-80 13:49

MACY11 30A(1052) 19-JUN-80 13:52 PAGE 200  
CROSS REFERENCE TABLE -- MACRO NAMES

C 16

SEQ 0197

.SMULT	1#		
.SPOWE	1#	834#	
.SRAND	1#		
.SRDDE	1#		
.SRDOC	1#	834#	7917
.SREAD	1#	834#	7661
.SR2AZ	1#		
.SSAVE	1#	834#	7970
.SSB2D	1#		
.SSB2O	1#		
.SSCOP	1#	834#	7109
.SSIZE	1#	834#	7032
.SSUPR	1#		
.STRAP	1#	834#	8047
.STYPB	1#		
.STYPD	1#	834#	7594
.STYPE	1#	834#	7438
.STYPO	1#	834#	7517
.S4OCA	1#		
.1170	1#		

. ABS. 065034 000

ERRORS DETECTED: 0

DSKZ:CZR6CD.BIN,DSKZ:CZR6CD.LST/CRF/SOL/NL:TOC=DSKZ:CZR6CD.SML,CZR6CD.P11  
RUN-TIME: 73 107 9 SECONDS  
RUN-TIME RATIO: 351/190=1.8  
CORE USED: 41K (81 PAGES)