

RK611  
RK06, RK07

RK611 DSKLS PRT 3  
CZR6CE0

AH-9106E-MC  
FICHE 1 OF 1

MAR 1982  
COPYRIGHT © 76-81  
MADE IN USA





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

.REM % IDENTIFICATION

PRODUCT CODE: AC-9104E-MC  
PRODUCT NAME: CZR6CEO RK611 DSKLS PRT3  
DATE: AUGUST 10 1981  
MAINTAINER: DIAGNOSTIC GROUP  
AUTHOR: BRIAN LEBLANC

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) 1976,1981 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88

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

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 N

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 (16K CORE MEMORY)

CONSOLE TERMINAL

DECTAPE, PAPER TAPE READER, OR DECDISK

RK611 CONTROLLER

2.2 PRELIMINARY PROGRAMS

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 1 (CZR6A)

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 2 (CZR6B)

ON-EMERGENCY STANT MEMORY AND UNIBUS PARITY ERROR DETECTION

89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144

### 3.0 OPERATING PROCEDURES

#### 3.1 LOADING PROCEDURE

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

#### 3.2 STARTING PROCEDURE

LOCATION 200 - START PROGRAM

LOCATION 204 - RESTART PROGRAM

LOCATION 214 - REQUEST BUS ADDRESS, VECTOR ADDRESS, AND PRIORITY MODIFICATION

#### 3.3 OPTIONAL SWITCH SETTINGS

SW15 - HALT PROGRAM  
SW14 - LOOP ON TEST  
SW13 - INHIBIT ERROR TYPE OUT  
SW12 - ABORT AFTER 20 ERRORS  
SW11 - INHIBIT ITERATION COUNT  
SW10 - BELL ON ERROR  
SW9 - LOOP ON ERROR  
SW8 - LOOP ON TEST IN SWITCHES 0-7

#### 3.5 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



145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200

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.

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

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.

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 ANY 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

257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312

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 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 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.



313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368

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  
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

369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424

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, HEADER 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.

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

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:

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 C.



481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536

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  
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

537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592

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:

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)

593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648

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  
022222  
111111

MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX MGD 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



649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704

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.  
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

705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760

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 ~~W~~LD 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, SPECIFY 66 WORDS. MAKE SURE ALL 26 SECTORS ARE WRITTEN CORRECTLY.

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 THE 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 ON 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 THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO A RK06, IN 24 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE 0.

761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816

CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN OFF  
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 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.

6.0 ERROR REPORTING

THE GENERAL FORMAT OF ERROR REPORTS IS:

OPERATION DESCRIPTION AND ERROR DESCRIPTION  
TEST ERROR  
NUM PC  
XXXXXX YYYYYY  
EXPECT ACTUAL OTHER PERTENANT  
REG REG INFORMATION  
ZZZZZZ WWWWAA AAAAAA

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

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

THESE FIELDS (ALL VALUES GIVEN IN OCTAL) ARE:

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

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

THE 'BIT COUNT' REPORTED IS INITIALIZED AT THE START OF EACH FIELD

817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830

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.

%

```
831 ; *** REV 003 ***
832 .NLIST CND,MD,MC
833 .LIST ME
834 .ENABL ABS,AMA
835 167400 $SWR= 167400
836 000001 $TN= 1
837 .TITLE CZR6CEO RK611 DSKLS CTRL PRT3
838 ;*COPYRIGHT (C) 1976,1981
839 ;*DIGITAL EQUIPMENT CORP.
840 ;*MAYNARD, MASS. 01754
841 ;*
842 ;*
843 ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
844 ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
845 ;*
846 .SBTTL OPERATIONAL SWITCH SETTINGS
847 ;*
848 ;* SWITCH USE
849 ;* -----
850 ;* 15 HALT ON ERROR
851 ;* 14 LOOP ON TEST
852 ;* 13 INHIBIT ERROR TYPEOUTS
853 ;* 12 ABORT PROGRAM AFTER 20 ERRORS
854 ;* 11 INHIBIT ITERATIONS
855 ;* 10 BELL ON ERROR
856 ;* 9 LOOP ON ERROR
857 ;* 8 LOOP ON TEST IN SWR<7:0>
858 .SBTTL BASIC DEFINITIONS
859 ;*
860 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
861 001100 $STACK= 1100
862 .EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
863 .EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
864 ;*
865 ;*MISCELLANEOUS DEFINITIONS
866 000011 HT= 11 ;;CODE FOR HORIZONTAL TAB
867 000012 LF= 12 ;;CODE FOR LINE FEED
868 000015 CR= 15 ;;CODE FOR CARRIAGE RETURN
869 000200 CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
870 177776 PS= 177776 ;;PROCESSOR STATUS WORD
871 .EQUIV PS,PSW
872 177774 $TKLMI= 177774 ;;STACK LIMIT REGISTER
873 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
874 177570 DSWR= 177570 ;;HARDWARE SWITCH REGISTER
875 177570 DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
876 ;*
877 ;*GENERAL PURPOSE REGISTER DEFINITIONS
878 000000 R0= %0 ;;GENERAL REGISTER
879 000001 R1= %1 ;;GENERAL REGISTER
880 000002 R2= %2 ;;GENERAL REGISTER
881 000003 R3= %3 ;;GENERAL REGISTER
882 000004 R4= %4 ;;GENERAL REGISTER
883 000005 R5= %5 ;;GENERAL REGISTER
884 000006 R6= %6 ;;GENERAL REGISTER
885 000007 R7= %7 ;;GENERAL REGISTER
886 000006 SP= %6 ;;STACK POINTER
```

```
887      000007      PC=      %7      ;;PROGRAM COUNTER
888
889      ;*PRIORITY LEVEL DEFINITIONS
890      000000      PR0=      0      ;;PRIORITY LEVEL 0
891      000040      PR1=      40     ;;PRIORITY LEVEL 1
892      000100      PR2=      100    ;;PRIORITY LEVEL 2
893      000140      PR3=      140    ;;PRIORITY LEVEL 3
894      000200      PR4=      200    ;;PRIORITY LEVEL 4
895      000240      PR5=      240    ;;PRIORITY LEVEL 5
896      000300      PR6=      300    ;;PRIORITY LEVEL 6
897      000340      PR7=      340    ;;PRIORITY LEVEL 7
898
899      ;*'SWITCH REGISTER' SWITCH DEFINITIONS
900      100000      SW15=     100000
901      040000      SW14=     40000
902      020000      SW13=     20000
903      010000      SW12=     10000
904      004000      SW11=     4000
905      002000      SW10=     2000
906      001000      SW09=     1000
907      000400      SW08=     400
908      000200      SW07=     200
909      000100      SW06=     100
910      000040      SW05=     40
911      000020      SW04=     20
912      000010      SW03=     10
913      000004      SW02=     4
914      000002      SW01=     2
915      000001      SW00=     1
916      .EQUIV      SW09,SW9
917      .EQUIV      SW08,SW8
918      .EQUIV      SW07,SW7
919      .EQUIV      SW06,SW6
920      .EQUIV      SW05,SW5
921      .EQUIV      SW04,SW4
922      .EQUIV      SW03,SW3
923      .EQUIV      SW02,SW2
924      .EQUIV      SW01,SW1
925      .EQUIV      SW00,SW0
926
927      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
928      100000      BIT15=    100000
929      040000      BIT14=    40000
930      020000      BIT13=    20000
931      010000      BIT12=    10000
932      004000      BIT11=    4000
933      002000      BIT10=    2000
934      001000      BIT09=    1000
935      000400      BIT08=    400
936      000200      BIT07=    200
937      000100      BIT06=    100
938      000040      BIT05=    40
939      000020      BIT04=    20
940      000010      BIT03=    10
941      000004      BIT02=    4
942      000002      BIT01=    2
```



```
943      000001      BIT00= 1
944      .EQUIV BIT09,BIT9
945      .EQUIV BIT08,BIT8
946      .EQUIV BIT07,BIT7
947      .EQUIV BIT06,BIT6
948      .EQUIV BIT05,BIT5
949      .EQUIV BIT04,BIT4
950      .EQUIV BIT03,BIT3
951      .EQUIV BIT02,BIT2
952      .EQUIV BIT01,BIT1
953      .EQUIV BIT00,BIT0
954
955      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
956      000004      ERRVEC= 4          ;; TIME OUT AND OTHER ERRORS
957      000010      RESVEC= 10         ;; RESERVED AND ILLEGAL INSTRUCTIONS
958      000014      TBITVEC=14        ;; "T" BIT
959      000014      TRTVEC= 14         ;; TRACE TRAP
960      000014      BPTVEC= 14        ;; BREAKPOINT TRAP (BPT)
961      000020      IOTVEC= 20         ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
962      000024      PWRVEC= 24         ;; POWER FAIL
963      000030      EMTVEC= 30         ;; EMULATOR TRAP (EMT) **ERROR**
964      000034      TRAPVEC=34        ;; "TRAP" TRAP
965      000060      TKVEC= 60          ;; TTY KEYBOARD VECTOR
966      000064      TPVEC= 64          ;; TTY PRINTER VECTOR
967      000240      PIRQVEC=240        ;; PROGRAM INTERRUPT REQUEST VECTOR
968      .SBTTL MEMORY MANAGEMENT DEFINITIONS
969
970      ;*KT11 VECTOR ADDRESS
971
972      000250      MMVEC= 250
973
974      ;*KT11 STATUS REGISTER ADDRESSES
975
976      177572      SR0= 177572
977      177574      SR1= 177574
978      177576      SR2= 177576
979      172516      SR3= 172516
980
981      ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
982
983      172300      KIPDR0= 172300
984      172302      KIPDR1= 172302
985      172304      KIPDR2= 172304
986      172306      KIPDR3= 172306
987      172310      KIPDR4= 172310
988      172312      KIPDR5= 172312
989      172314      KIPDR6= 172314
990      172316      KIPDR7= 172316
991
992      ;*KERNEL "I" PAGE ADDRESS REGISTERS
993
994      172340      KIPAR0= 172340
995      172342      KIPAR1= 172342
996      172344      KIPAR2= 172344
997      172346      KIPAR3= 172346
998      172350      KIPAR4= 172350
```

```
999          172352      KIPAR5= 172352
1000         172354      KIPAR6= 172354
1001         172356      KIPAR7= 172356
1002
1003         000114      MEMVEC= 114          ;MEMORY PARITY VECTOR
1004         172100      MEMBAS= 172100      ;MEM PARITY OPTION
1005         000004      WR.PAR= 4           ;WRITE BAD PARITY
1006         000001      PAR.EN= 1           ;ENABLE PARITY ENABLE
1007         120210      AVECT1= 120210     ;DEFINE RK611 VECTOR ADDRESS
1008         000005      APRIOR= 5           ;DEFINE RK611 PRIORITY
1009         177440      ABASE= 177440      ;DEFINE BASE OF RK611 REGISTERS
1010
1011          .SBTTL  RK611 CONTROLLER REGISTER DEFINITION
1012
1013         000000      RKCS1= 0            ;CONTROL AND STATUS REGISTER 1
1014         000002      RKWC= 2            ;WORD COUNT REGISTER
1015         000004      RKBA= 4            ;BUS ADDRESS REGISTER
1016         000006      RKDA= 6            ;DESIRED TRACK SECTOR REGISTER
1017         000010      RKCS2= 10           ;CONTROL AND STATUS REGISTER 2
1018         000012      RKDS= 12           ;DRIVE STATUS REGISTER
1019         000014      RKER= 14           ;ERROR REGISTER
1020         000016      RKASOF= 16          ;ATTENTION SUMMARY AND OFFSET REGISTER
1021         000020      RKDCYL= 20          ;DESIRED CYLINDER REGISTER
1022         000024      RKDB= 24           ;DATA BUFFER
1023         000026      RKMR1= 26          ;MAINTENANCE REGISTER 1
1024         000034      RKMR2= 34          ;MAINTENANCE REGISTER 2
1025         000036      RKMR3= 36          ;MAINTENANCE REGISTER 3
1026         000030      RKECPS= 30         ;ECC POSITION INFORMATION
1027         000032      RKECPT= 32         ;ECC PATTERN INFORMATION
1028         000022      RKSPAR= 22         ;SPARE REGISTER
1029
1030          .SBTTL  DRIVE COMMANDS
1031
1032         000001      SELDRV= 01          ;SELECT DRIVE
1033         000003      PACK= 03           ;PACK ACKNOWLEDGE
1034         000005      CLEAR= 05          ;DRIVE CLEAR
1035         000007      UNLOAD= 07         ;UNLOAD
1036         000011      SRTSPL= 11         ;START SPINDLE
1037         000013      RECAL= 13          ;RECALIBRATE
1038         000015      OFFSET= 15         ;OFFSET
1039         000017      SEEK= 17           ;SEEK
1040         000021      RDDATA= 21         ;READ DATA
1041         000023      WRDATA= 23         ;WRITE DATA
1042         000025      RDHEAD= 25         ;READ HEADER
1043         000027      WRHEAD= 27         ;WRITE HEADER AND DATA
1044         000031      WRTCHK= 31         ;WRITE CHECK
1045         000300      INTR= 300          ;GENERATE INTERRUPT TO CPU
1046
1047          .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
1048
1049         000001      GO= BIT0            ;GO BIT
1050         000100      IE= BIT6            ;INTERRUPT ENABLE
1051         000200      RDY= BIT7           ;CONTROLLER READY
1052         000400      BA16= BIT8          ;BUS ADDRESS BIT 16
1053         001000      BA17= BIT9          ;BUS ADDRESS BIT 17
1054         002000      CDT= BIT10         ;CONTROLLER DRIVE TYPE (0=RK06)
```

1055	004000	CTO= BIT11	: CONTROLLER TIMED OUT WAITING FOR
1056			: DRIVE RESPONSE
1057	010000	CFMT= BIT12	: CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
1058	020000	SPAR= BIT13	: DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1059	040000	DI= BIT14	: DRIVE INTERRUPT
1060	100000	CERR= BIT15	: CONTROLLER ERROR
1061	100000	CCLR= BIT15	: CONTROLLER CLEAR

.SBTTL CONTROL AND STATUS REGISTER 2 BITS

1065	000007	DRVMSK= 7	: MASK FOR DRIVE SELECTION CODE
1066	000010	RLS= BIT3	: DESELECT OR RELEASE DRIVE IN BITS 0-2
1067	000020	BAI= BIT4	: BUS ADDRESS INCREMENT INHIBIT
1068	000040	SCLR= BIT5	: CLEAR CONTROLLER AND ALL DRIVES
1069	000100	IR= BIT6	: INPUT READY
1070	000200	OR= BIT7	: OUTPUT READY
1071	000400	UFE= BIT8	: UNIT FIELD ERROR
1072	001000	MDS= BIT9	: MULTIPLE DRIVE SELECT
1073	002000	PGE= BIT10	: PROGRAMMING ERROR
1074	004000	NEM= BIT11	: NON-EXISTENT MEMORY
1075	010000	NED= BIT12	: NON-EXISTENT DRIVE
1076	020000	UPE= BIT13	: UNIBUS PARITY ERROR
1077	040000	WCE= BIT14	: WRITE CHECK ERROR
1078	100000	DLT= BIT15	: DATA LATE ERROR

.SBTTL ERROR REGISTER BIT DEFINITION

1082	000001	ILF= BIT0	: ILLEGAL FUNCTION CODE
1083	000002	SKI= BIT1	: SEEK INCOMPLETE
1084	000004	NXF= BIT2	: NON-EXECUTABLE DRIVE FUNCTION
1085	000010	DRPAR= BIT3	: DRIVE DETECTED DRIVE BUS PARITY ERROR
1086	000020	FMTE= BIT4	: FORMAT ERROR
1087	000040	DTYE= BIT5	: DRIVE TYPE ERROR
1088	000100	ECH= BIT6	: ECC HARD
1089	000200	BSE= BIT7	: BAD SECTOR ERROR
1090	000400	HVRC= BIT8	: HEADER VRC ERROR
1091	001000	COE= BIT9	: CYLINDER ADDRESS OVERFLOW ERROR
1092	002000	IDAE= BIT10	: INVALID DISK ADDRESS ERROR
1093	004000	WLE= BIT11	: WRITE LOCK ERROR
1094	010000	DTE= BIT12	: DRIVE TIMING ERROR
1095	020000	OPI= BIT13	: OPERATION (SEARCH) INCOMPLETE
1096	040000	UNS= BIT14	: DRIVE UNSAFE
1097	100000	DCK= BIT15	: DATA CHECK

.SBTTL STATUS REGISTER BIT DEFINITION

1101	000001	DRA= BIT0	: DRIVE AVAILABLE (CONTROLLER IS SET IF : THIS BIT IS RESET)
1103	000004	OFST= BIT2	: DRIVE OFFSET
1104	000010	ACLO= BIT3	: AC LOW
1105	000020	SPDLSS= BIT4	: SPEED LOSS
1106	000040	DROT= BIT5	: DRIVE OFF TRACK
1107	000100	VV= BIT6	: VOLUME VALID
1108	000200	DRDY= BIT7	: DRIVE READY
1109	000400	DDT= BIT8	: DRIVE TYPE (0=RK06)
1110	004000	WRL= BIT11	: WRITE LOCK

```

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

1167 000114 043650  
1168 000116 000340  
1169 00'000  
1170  
1171  
1172  
1173  
1174  
1175 001000  
1176 000024 000024  
1177 000024 000200  
1178 000044 000044  
1179 000044 001000  
1180 001000  
1181  
1182  
1183  
1184  
1185 001000  
1186 001000 000000  
1187 001002 001214  
1188 001004 000000  
1189 001006 000000  
1190 001010 000000  
1191 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)  
\$TSTM: .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  
.WORD \$ETEND-\$MAIL/2 ::LENGTH MAILBOX-ETABLE(WORDS)

1192  
1193  
1194  
1195  
1196  
1197  
1198 001100  
1199 001100  
1200 001100 000000  
1201 001102 000  
1202 001103 000  
1203 001104 000000  
1204 001106 000000  
1205 001110 000000  
1206 001112 000000  
1207 001114 000  
1208 001115 001  
1209 001116 000000  
1210 001120 000000  
1211 001122 000000  
1212 001124 000000  
1213 001126 000000  
1214 001130 000000  
1215 001132 000000  
1216 001134 000  
1217 001135 000  
1218 001136 000000  
1219 001140 177570  
1220 001142 177570  
1221 001144 177560  
1222 001146 177562  
1223 001150 177564  
1224 001152 177566  
1225 001154 000  
1226 001155 002  
1227 001156 012  
1228 001157 000  
1229 001160 000000  
1230 001162 000000  
1231 001164 000000  
1232 001166 000000  
1233 001170 000000  
1234 001172 000000  
1235 001174 000000  
1236 001176 000000  
1237 001200 000000  
1238 001202 000000  
1239 001204 177607 000377  
1240 001210 077  
1241 001211 015  
1242 001212 000012  
1243  
1244  
1245  
1246  
1247

.SBTTL COMMON TAGS

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

.=1100

\$CMTAG: .WORD 0 :: START OF COMMON TAGS  
\$TSTNM: .BYTE 0 :: CONTAINS THE TEST NUMBER  
\$ERFLG: .BYTE 0 :: CONTAINS ERROR FLAG  
\$ICNT: .WORD 0 :: CONTAINS SUBTEST ITERATION COUNT  
\$LPADR: .WORD 0 :: CONTAINS SCOPE LOOP ADDRESS  
\$LPERR: .WORD 0 :: CONTAINS SCOPE RETURN FOR ERRORS  
\$ERTTL: .WORD 0 :: CONTAINS TOTAL ERRORS DETECTED  
\$ITEMB: .BYTE 0 :: CONTAINS ITEM CONTROL BYTE  
\$ERMAX: .BYTE 1 :: CONTAINS MAX. ERRORS PER TEST  
\$ERRPC: .WORD 0 :: CONTAINS PC OF LAST ERROR INSTRUCTION  
\$GDADR: .WORD 0 :: CONTAINS ADDRESS OF 'GOOD' DATA  
\$BDADR: .WORD 0 :: CONTAINS ADDRESS OF 'BAD' DATA  
\$GDDAT: .WORD 0 :: CONTAINS 'GOOD' DATA  
\$BDDAT: .WORD 0 :: CONTAINS 'BAD' DATA  
          .WORD 0 :: RESERVED--NOT TO BE USED  
          .WORD 0  
\$AUTOB: .BYTE 0 :: AUTOMATIC MODE INDICATOR  
\$INTAG: .BYTE 0 :: INTERRUPT MODE INDICATOR  
          .WORD 0  
SWR: .WORD DSWR :: ADDRESS OF SWITCH REGISTER  
DISP\_AY: .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



1248	001214		\$MAIL:		::APT MAILBOX
1249	001214	000000	\$MSGTY:	.WORD	AMSGTY ::MESSAGE TYPE CODE
1250	001216	000000	\$FATAL:	.WORD	AFATAL ::FATAL ERROR NUMBER
1251	001220	000000	\$TESTN:	.WORD	ATESTN ::TEST NUMBER
1252	001222	000000	\$PASS:	.WORD	APASS ::PASS COUNT
1253	001224	000000	\$DEVCT:	.WORD	ADEVCT ::DEVICE COUNT
1254	001226	000000	\$UNIT:	.WORD	AUNIT ::I/O UNIT NUMBER
1255	001230	000000	\$MSGAD:	.WORD	AMSGAD ::MESSAGE ADDRESS
1256	001232	000000	\$MSGLG:	.WORD	AMSGLG ::MESSAGE LENGTH
1257	001234		\$ETABLE:		::APT ENVIRONMENT TABLE
1258	001234	000	\$ENV:	.BYTE	AENV ::ENVIRONMENT BYTE
1259	001235	000	\$ENVM:	.BYTE	AENVM ::ENVIRONMENT MODE BITS
1260	001236	000000	\$SWREG:	.WORD	ASWREG ::APT SWITCH REGISTER
1261	001240	000000	\$USWR:	.WORD	AUSWR ::USER SWITCHES
1262	001242	000000	\$CPUOP:	.WORD	ACPUOP ::CPU TYPE,OPTIONS
1263			*		BITS 15-11=CPU TYPE
1264			*		11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1265			*		11/70=06,PDQ=07,Q=10
1266			*		BIT 10=REAL TIME CLOCK
1267			*		BIT 9=FLOATING POINT PROCESSOR
1268			*		BIT 8=MEMORY MANAGEMENT
1269	001244	000	\$MAMS1:	.BYTE	AMAMS1 ::HIGH ADDRESS,M.S. BYTE
1270	001245	000	\$MTYP1:	.BYTE	AMTYP1 ::MEM. TYPE,BLK#1
1271			*		MEM.TYPE BYTE -- (HIGH BYTE)
1272			*		900 NSEC CORE=001
1273			*		300 NSEC BIPOLAR=002
1274			*		500 NSEC MOS=003
1275	001246	000000	\$MADR1:	.WORD	AMADR1 ::HIGH ADDRESS,BLK#1
1276			*		MEM.LAST ADDR.=3 BYTES,THIS WURD AND LOW OF "TYPE" ABOVE
1277	001250	000	\$MAMS2:	.BYTE	AMAMS2 ::HIGH ADDRESS,M.S. BYTE
1278	001251	000	\$MTYP2:	.BYTE	AMTYP2 ::MEM. TYPE,BLK#2
1279	001252	000000	\$MADR2:	.WORD	AMADR2 ::MEM.LAST ADDRESS,BLK#2
1280	001254	000	\$MAMS3:	.BYTE	AMAMS3 ::HIGH ADDRESS,M.S.BYTE
1281	001255	000	\$MTYP3:	.BYTE	AMTYP3 ::MEM. TYPE,BLK#3
1282	001256	000000	\$MADR3:	.WORD	AMADR3 ::MEM.LAST ADDRESS,BLK#3
1283	001260	000	\$MAMS4:	.BYTE	AMAMS4 ::HIGH ADDRESS,M.S.BYTE
1284	001261	000	\$MTYP4:	.BYTE	AMTYP4 ::MEM. TYPE,BLK#4
1285	001262	000000	\$MADR4:	.WORD	AMADR4 ::MEM.LAST ADDRESS,BLK#4
1286	001264	120210	\$VECT1:	.WORD	AVECT1 ::INTERRUPT VECTOR#1,BUS PRIORITY#1
1287	001266	000000	\$VECT2:	.WORD	AVECT2 ::INTERRUPT VECTOR#2BUS PRIOR.TY#2
1288	001270	177440	\$BASE:	.WORD	ABASE ::BASE ADDRESS OF EQUIPMENT UNDER TEST
1289	001272	000000	\$DEVN:	.WORD	ADEVN ::DEVICE MAP
1290	001274	000000	\$CDW1:	.WORD	ACDW1 ::CONTROLLER DESCRIPTION WORD#1
1291	001276	000000	\$CDW2:	.WORD	ACDW2 ::CONTROLLER DESCRIPTION WORD#2
1292	001300		\$ETEND:		
1293			.MEXIT		

```
1294 .SBTTL ERROR POINTER TABLE
1295
1296 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1297 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1298 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1299 ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
1300 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1301
1302 ;*      EM          ;;POINTS TO THE ERROR MESSAGE
1303 ;*      DH          ;;POINTS TO THE DATA HEADER
1304 ;*      DT          ;;POINTS TO THE DATA
1305 ;*      DF          ;;POINTS TO THE DATA FORMAT
1306
1307
1308 001300 $ERRTB:
1309 :      ERROR 1:  ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER
1310 :                  CS? INCORRECT.
1311 001300 055405      EM200
1312 001302 061564      EM3000
1313 001304 050644      DT001
1314 001306 051306      DF001
1315 :      ERROR 2:  ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER
1316 :                  MESSAGE A INCORRECT.
1317 001310 055405      EM200
1318 001312 061627      EM3001
1319 001314 050644      DT001
1320 001316 051306      DF001
1321 :      ERROR 3:  ATTEMPTING TO CHECK SEFK MESSAGE FROM READ HEADER
1322 :                  MESSAGE B INCORRECT.
1323 001320 055405      EM200
1324 001322 061653      EM3002
1325 001324 050644      DT001
1326 001326 051306      DF001
1327 :      ERROR 4:  ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER
1328 :                  CS1 INCORRECT.
1329 001330 055467      EM201
1330 001332 061564      EM3000
1331 001334 050644      DT001
1332 001336 051306      DF001
1333 :      ERROR 5:  ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER
1334 :                  MESSAGE A INCORRECT.
1335 001340 055467      EM201
1336 001342 061627      E13001
1337 001344 050644      DT001
1338 001346 051306      DF001
1339 :      ERROR 6:  ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER
1340 :                  MESSAGE IS INCORRECT.
1341 001350 055467      EM201
1342 001352 061653      EM3002
1343 001354 050644      DT001
1344 001356 051306      DF001
1345 :      ERROR 7:  ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
1346 :                  CS1 INCORRECT
1347 001360 055552      EM202
1348 001362 061564      EM3000
1349 001364 050644      DT001
```

1350	001366	051306	DF001
1351	:	:	ERROR 10: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
1352	:	:	MESSAGE A INCORRECT.
1353	001370	055552	EM202
1354	001372	061627	EM3001
1355	001374	050644	DT001
1356	001376	051306	DF001
1357	:	:	ERROR 11: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM READ HEADER
1358	:	:	MESSAGE B INCORRECT.
1359	001400	055552	EM202
1360	001402	061653	EM3002
1361	001404	050644	DT001
1362	001406	051306	DF001
1363	:	:	ERROR 12: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1364	:	:	CS1 INCORRECT.
1365	001410	055643	EM203
1366	001412	061564	EM3000
1367	001414	050644	DT001
1368	001416	051306	DF001
1369	:	:	ERROR 13: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1370	:	:	MESSAGE A INCORRECT.
1371	001420	055643	EM203
1372	001422	061627	EM3001
1373	001424	050644	DT001
1374	001426	051306	DF001
1375	:	:	ERROR 14: ATTEMPTING TO CHECK DRIVE CLEAR MESSAGE FROM WRITE HEADER
1376	:	:	MESSAGE B INCORRECT.
1377	001430	055643	EM203
1378	001432	061653	EM3002
1379	001434	050644	DT001
1380	001436	051306	DF001
1381	:	:	ERROR 15: ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1382	:	:	CS1 INCORRECT AFTER SENDING DRIVE CLEAR.
1383	001440	055735	EM204
1384	001442	061677	EM3003
1385	001444	050664	DT015
1386	001446	051332	DF015
1387	:	:	ERROR 16: ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1388	:	:	CS1 INCORRECT AFTER DATA SIMULATION.
1389	001450	055735	EM204
1390	001452	061747	EM3004
1391	001454	050664	DT015
1392	001456	051332	DF015
1393	:	:	ERROR 17: ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1394	:	:	MAINT REG. 1 INCORRECT DURING DATA SIMULATION (SECTOR PULSE)
1395	001460	055735	EM204
1396	001462	062021	EM3005
1397	001464	050674	DT017
1398	001466	051356	DF017
1399	:	:	ERROR 20: ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT
1400	:	:	MAINT REG. 1 INCORRECT DURING DATA SIMULATION (INDEX PULSE)
1401	001470	055735	EM204
1402	001472	062123	EM3006
1403	001474	050674	DT017
1404	001476	051356	DF017
1405	:	:	ERROR 21: ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT

1406	:	MAINT REG. 1 INCORRECT DURING DATA SIMULATION (NO INDEX OR SECTOR)	
1407	001500	055735	EM204
1408	001502	062224	EM3007
1409	001504	050674	DT017
1410	001506	051356	DF017
1411	:	ERROR 22: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1412	:	CS1 INCORRECT AFTER SENDING DRIVE CLEAR	
1413	001510	056023	EM205
1414	001512	061677	EM3003
1415	001514	050712	DT022
1416	001516	051402	DF022
1417	:	ERROR 23: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1418	:	BUS ADD INCORRECT AFTER SENDING DRIVE CLEAR.	
1419	001520	056023	EM205
1420	001522	062345	EM3008
1421	001524	050712	DT022
1422	001526	051402	DF022
1423	:	ERROR 24: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1424	:	WORD COUNT INCORRECT AFTER SENDING DRIVE CLEAR	
1425	001530	056023	EM205
1426	001532	062425	EM3009
1427	001534	050712	DT022
1428	001536	051402	DF022
1429	:	ERROR 25: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1430	:	MAINT REG 1 INCORRECT DURING DATA SIMULATION (NO INDEX OR SECTOR)	
1431	001540	056023	EM205
1432	001542	062723	EM3014
1433	001544	050732	DT025
1434	001546	051426	DF025
1435	:	ERROR 26: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1436	:	CS1 CHANGED DURING COMMAND EXECUTION	
1437	001550	056023	EM205
1438	001552	062504	EM3010
1439	001554	050712	DT022
1440	001556	051402	DF022
1441	:	ERROR 27: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1442	:	BUS ADDRESS CHANGED BEFORE INDEX PULSE	
1443	001560	056023	EM205
1444	001562	062551	EM3011
1445	001564	050712	DT022
1446	001566	051402	DF022
1447	:	ERROR 30: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1448	:	WORD COUNT CHANGED BEFORE INDEX PULSE	
1449	001570	056023	EM205
1450	001572	062620	EM3012
1451	001574	050712	DT022
1452	001576	051402	DF022
1453	:	ERROR 31: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1454	:	CS1 CHANGED AFTER INDEX PULSE	
1455	001600	056023	EM205
1456	001602	062666	EM3013
1457	001604	050744	DT031
1458	001606	051452	DF031
1459	:	ERROR 32: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT	
1460	:	BUS ADDRESS CHANGED AFTER INDEX PULSE.	
1461	001610	056023	EM205

1462	001612	062774	EM3015
1463	001614	050744	DT031
1464	001616	051452	DF031
1465	:	:	ERROR 33: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1466	:	:	WORD COUNT CHANGED AFTER INDEX PULSE
1467	001620	056023	EM205
1468	001622	063042	EM3016
1469	001624	050744	DT031
1470	001626	051452	DF031
1471	:	:	ERROR 34: ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT
1472	:	:	MAINT REG 1 INCORRECT AFTER SECTOR PULSE
1473	001630	056023	EM205
1474	001632	063422	EM3025
1475	001634	050744	DT031
1476	001636	051452	DF031
1477	:	:	ERROR 35: ATTEMPTING AN NPR READ OF ONE WORD
1478	:	:	CS1 INCORRECT
1479	001640	056111	EM206
1480	001642	061564	EM3000
1481	001644	050770	DT035
1482	001646	051476	DF035
1483	:	:	ERROR 36: ATTEMPTING AN NPR READ OF ONE WORD
1484	:	:	CS2 INCORRECT
1485	001650	056111	EM206
1486	001652	063107	EM3018
1487	001654	050770	DT035
1488	001656	051476	DF035
1489	:	:	ERROR 37: ATTEMPTING AN NPR READ OF ONE WORD
1490	:	:	BUS ADDRESS INCORRECT
1491	001660	056111	EM206
1492	001662	063153	EM3019
1493	001664	050770	DT035
1494	001666	051476	DF035
1495	:	:	ERROR 40: ATTEMPTING AN NPR READ OF ONE WORD
1496	:	:	WORD COUNT REG INCORRECT
1497	001670	056111	EM206
1498	001672	063201	EM3020
1499	001674	050770	DT035
1500	001676	051476	DF035
1501	:	:	ERROR 41: ATTEMPTING AN NPR READ OF ONE WORD
1502	:	:	WORD READ INCORRECT
1503	001700	056111	EM206
1504	001702	063232	EM3021
1505	001704	051014	DT041
1506	001706	051522	DF041
1507	:	:	ERROR 42: ATTEMPTING AN NPR READ OF ONE WORD
1508	:	:	CS1 INCORRECT AFTER READING DATA BUFFER
1509	001710	056111	EM206
1510	001712	063256	EM3022
1511	001714	051024	DT042
1512	001716	051546	DF042
1513	:	:	ERROR 43: ATTEMPTING AN NPR READ OF ONE WORD
1514	:	:	CS2 INCORRECT AFTER READING DATA BUFFER
1515	001720	056111	EM206
1516	001722	063326	EM3023
1517	001724	051024	DT042

CZR6CEO RK611 DSKLS CTRL PRT3  
CZR6CE.P11 27-AUG-81 10:24

MACV11 30(1046) 28-AUG-81 10:35 E 3 PAGE 31  
ERROR POINTER TABLE

SEQ 0030

1518	001726	051546	DF042
1519			ERROR 44: ATTEMPTING AN NPR READ OF ONE WORD
1520			CS1 INCORRECT
1521	001730	056154	EM207
1522	001732	061564	EM3000
1523	001734	050770	DT035
1524	001736	051476	DF035
1525			ERROR 45: ATTEMPTING AN NPR READ
1526			CS2 INCORRECT
1527	001740	056154	EM207
1528	001742	063107	EM3018
1529	001744	050770	DT035
1530	001746	051476	DF035
1531			ERROR 46: ATTEMPTING AN NPR READ
1532			BUS ADDRESS INCORRECT



1533	001750	056154	EM207
1534	001752	063153	EM3019
1535	001754	050770	DT035
1536	001756	051476	DF035
1537	:	:	ERROR 47: ATTEMPTING AN NPR READ
1538	:	:	WORD COUNT INCORRECT
1539	001760	056154	EM207
1540	001762	063201	EM3020
1541	001764	050770	DT035
1542	001766	051476	DF035
1543	:	:	ERROR 50: ATTEMPTING NPR READ CHECKING ZERO DETECT
1544	:	:	CS1 INCORRECT
1545	001770	056203	EM208
1546	001772	061564	EM3000
1547	001774	050770	DT035
1548	001776	051476	DF035
1549	:	:	ERROR 51: ATTEMPTING NPR READ CHECKING ZERO DETECT
1550	:	:	CS2 INCORRECT
1551	002000	056203	EM208
1552	002002	063107	EM3018
1553	002004	050770	DT035
1554	002006	051476	DF035
1555	:	:	ERROR 52: ATTEMPTING NPR READ CHECKING ZERO DETECT
1556	:	:	BUS ADDRESS INCORRECT
1557	002010	056203	EM208
1558	002012	063153	EM3019
1559	002014	050770	DT035
1560	002016	051476	DF035
1561	:	:	ERROR 53: ATTEMPTING NPR READ CHECKING ZERO DETECT
1562	:	:	WORD COUNT INCORRECT
1563	002020	056203	EM208
1564	002022	063201	EM3020
1565	002024	050770	DT035
1566	002026	051476	DF035
1567	:	:	ERROR 54: ATTEMPTING NPR READ
1568	:	:	DATA BUFFER INCORRECT
1569	002030	056154	EM207
1570	002032	063232	EM3021
1571	002034	051040	DT054
1572	002036	051572	DF054
1573	:	:	ERROR 55: ATTEMPTING NPR READ
1574	:	:	CS1 INCORRECT AFTER READING DATA BUFFER
1575	002040	056154	EM207
1576	002042	063256	EM3022
1577	002044	051052	DT055
1578	002046	051616	DF055
1579	:	:	ERROR 56: ATTEMPTING NPR READ
1580	:	:	CS2 INCORRECT AFTER READING DATA BUFFER
1581	002050	056154	EM207
1582	002052	063326	EM3023
1583	002054	051052	DT055
1584	002056	051616	DF055
1585	:	:	ERROR 57: ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1586	:	:	INHIBIT CS1 INCORRECT
1587	002060	056254	EM209
1588	002062	061564	EM3000

1589	002064	050770	DT035	
1590	002066	051476	DF035	
1591			ERROR 60:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1592				INHIBIT CS2 INCORRECT
1593	002070	056254	EM209	
1594	002072	063107	EM3018	
1595	002074	050770	DT035	
1596	002076	051476	DF035	
1597			ERROR 61:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1598				INHIBIT BUS ADDRESS INCORRECT
1599	002100	056254	EM209	
1600	002102	063153	EM3019	
1601	002104	050770	DT035	
1602	002106	051476	DF035	
1603			ERROR 62:	ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT
1604				INHIBIT WORD COUNT INCORRECT
1605	002110	056254	EM209	
1606	002112	063201	EM3020	
1607	002114	050770	DT035	
1608	002116	051476	DF035	
1609			ERROR 63:	ATTEMPTING NPR READ WITH IBA TO CHECK ZERO
1610				DETECT-CS1 INCORRECT
1611	002120	056343	EM210	
1612	002122	061564	EM3000	
1613	002124	050770	DT035	
1614	002126	051476	DF035	
1615			ERROR 64:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1616				DETECT-CS2 INCORRECT
1617	002130	056343	EM210	
1618	002132	063107	EM3018	
1619	002134	050770	DT035	
1620	002136	051476	DF035	
1621			ERROR 65:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1622				DETECT-BUS ADDRESS INCORRECT
1623	002140	056343	EM210	
1624	002142	063153	EM3019	
1625	002144	050770	DT035	
1626	002146	051476	DF035	
1627			ERROR 66:	ATTEMPTING NPR READ WITH BAI TO CHECK ZERO
1628				DETECT-WORD COUNT INCORRECT
1629	002150	056343	EM210	
1630	002152	063201	EM3020	
1631	002154	050770	DT035	
1632	002156	051476	DF035	
1633			ERROR 67:	ATTEMPTING NPR READ WITH BUS ADDRESS INHIBIT
1634				INCREMENT-DATA BUFFER INCORRECT
1635	002160	056254	EM209	
1636	002162	063232	EM3021	
1637	002164	051040	DT054	
1638	002166	051572	DF054	
1639			ERROR 70:	ATTEMPTING NPR READ WITH BUS ADDRESS INHIBIT
1640				INCREMENT-CS1 INCORRECT AFTER READING DATA BUFFER
1641	002170	056254	EM209	
1642	002172	063256	EM3022	
1643	002174	051052	DT055	
1644	002176	051616	DF055	

1645	:	ERROR 71: ATTEMPTING NPR READ WITH ADDRESS BUFFER INHIBIT
1646	:	INCREMENT-CS2 INCORRECT AFTER READING DATA BUFFER
1647	002200 056254	EM209
1648	002202 063326	EM3023
1649	002204 051052	DT055
1650	002206 051616	DF055
1651	:	ERROR 72: ATTEMPTING TO FORCE NON-EXISTANT MFMORY
1652	:	CS1 INCORRECT
1653	002210 056457	EM211
1654	002212 061564	EM3000
1655	002214 051070	DT072
1656	002216 051642	DF072
1657	:	ERROR 73: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1658	:	CS2 INCORRECT
1659	002220 056457	EM211
1660	002222 063107	EM3018
1661	002224 051070	DT072
1662	002226 051642	DF072
1663	:	ERROR 74: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1664	:	ERROR REG INCORRECT
1665	002230 056457	EM211
1666	002232 063376	EM3024
1667	002234 051070	DT072
1668	002236 051642	DF072
1669	:	ERROR 75: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1670	:	BUS ADDRESS INCORRECT
1671	002240 056457	EM211
1672	002242 063153	EM3019
1673	002244 051070	DT072
1674	002246 051642	DF072
1675	:	ERROR 76: ATTEMPTING TO FORCE NON-EXISTENT MEMORY
1676	:	WORD COUNT INCORRECT
1677	002250 056457	EM211
1678	002252 063201	EM3020
1679	002254 051070	DT072
1680	002256 051642	DF072
1681	:	ERROR 77: ATTEMPTING TO CLEAR NON-EXISTENT MEMORY
1682	:	CS1 INCORRECT
1683	002260 056527	EM212
1684	002262 061564	EM3000
1685	002264 051120	DT077
1686	002266 051676	DF077
1687	:	ERROR 100: ATTEMPTING TO CLEAR NON-EXISTENT MEMORY
1688	:	CS2 INCORRECT
1689	002270 056527	EM212
1690	002272 063107	EM3018
1691	002274 051120	DT077
1692	002276 051676	DF077
1693	:	ERROR 101: TESTING EXTENDED MEMORY ADDRESSING BITS
1694	:	CS1 INCORRECT
1695	002300 056577	EM213
1696	002302 061564	EM3000
1697	002304 050770	DT035
1698	002306 051476	DF035
1699	:	ERROR 102: TESTING EXTENDED MEMORY ADDRESSING BITS
1700	:	CS2 INCORRECT

1701	002310	056577	EM213
1702	002312	063107	EM3018
1703	002314	050770	DT035
1704	002316	051476	DF035
1705	:	:	ERROR 103: TESTING EXTENDED MEMORY ADDRESSING BITS
1706	:	:	BUS ADDRESS INCORRECT
1707	002320	056577	EM213
1708	002322	063153	EM3019
1709	002324	050770	DT035
1710	002326	051476	DF035
1711	:	:	ERROR 104: TESTING EXTEND MEMORY ADDRESSING BITS
1712	:	:	WORD COUNT INCORRECT
1713	002330	056577	EM213
1714	002332	063201	EM3020
1715	002334	050770	DT035
1716	002336	051476	DF035
1717	:	:	ERROR 105: TESTING EXTENDED MEMORY ADDRESSING BITS
1718	:	:	DATA BUFFER INCORRECT
1719	002340	056577	EM213
1720	002342	063232	EM3021
1721	002344	051014	DT041
1722	002346	051522	DF041
1723	:	:	ERROR 106: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1724	:	:	CS1 INCORRECT
1725	002350	056647	EM214
1726	002352	061564	EM3000
1727	002354	051070	DT072
1728	002356	051642	DF072
1729	:	:	ERROR 107: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1730	:	:	CS2 INCORRECT
1731	002360	056647	EM214
1732	002362	063107	EM3018
1733	002364	051070	DT072
1734	002366	051642	DF072
1735	:	:	ERROR 110: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1736	:	:	BUS ADDRESS INCORRECT
1737	002370	056647	EM214
1738	002372	063153	EM3019
1739	002374	051070	DT072
1740	002376	051642	DF072
1741	:	:	ERROR 111: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1742	:	:	WORD COUNT INCORRECT
1743	002400	056647	EM214
1744	002402	063201	EM3020
1745	002404	051070	DT072
1746	002406	051642	DF072
1747	:	:	ERROR 112: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1748	:	:	CS1 INCORRECT
1749	002410	056717	EM215
1750	002412	061564	EM3000
1751	002414	050770	DT035
1752	002416	051476	DF035
1753	:	:	ERROR 113: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1754	:	:	CS2 INCORRECT
1755	002420	056717	EM215
1756	002422	063107	EM3018

1757	002424	050770	DT035
1758	002426	051476	DF035
1759	:	:	ERROR 114: ATTEMPTING NPR READ OR LOCATION PRIOR TO BAD PARITY
1760	:	:	BUS ADDRESS INCORRECT
1761	002430	056717	EM215
1762	002432	063153	EM3019
1763	002434	050770	DT035
1764	002436	051476	DF035
1765	:	:	ERROR 115: ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY
1766	:	:	WORD COUNT INCORRECT
1767	002440	056717	EM215
1768	002442	063107	EM3018
1769	002444	050770	DT035
1770	002446	051476	DF035
1771	:	:	ERROR 116: ATTEMPTING TO FORCE UNIBUS PARITY ERROR
1772	:	:	ERROR REG INCORRECT
1773	002450	056647	EM214
1774	002452	063376	EM3024
1775	002454	051070	DT072
1776	002456	051642	DF072
1777	:	:	ERROR 117: ATTEMPTING TO CLEAR UNIBUS PARITY ERROR
1778	:	:	CS1 INCORRECT
1779	002460	057003	EM216
1780	002462	061564	EM3000
1781	002464	051120	DT077
1782	002466	051676	DF077
1783	:	:	ERROR 120: ATTEMPTING TO CLEAR UNIBUS PARITY ERROR
1784	:	:	CS2 INCORRECT
1785	002470	057003	EM216
1786	002472	063107	EM3018
1787	002474	051120	DT077
1788	002476	051676	DF077
1789	:	:	ERROR 121: ATTEMPTING 18 BIT NPR READ
1790	:	:	CS1 INCORRECT
1791	002500	057053	EM217
1792	002502	061564	EM3000
1793	002504	050770	DT035
1794	002506	051476	DF035
1795	:	:	ERROR 122: ATTEMPTING 18 BIT NPR READ
1796	:	:	CS2 INCORRECT
1797	002510	057053	EM217
1798	002512	063107	EM3018
1799	002514	050770	DT035
1800	002516	051476	DF035
1801	:	:	ERROR 123: ATTEMPTING 18 BIT NPR READ
1802	:	:	BUS ADDRESS INCORRECT
1803	002520	057053	EM217
1804	002522	063153	EM3019
1805	002524	050770	DT035
1806	002526	051476	DF035
1807	:	:	ERROR 124: ATTEMPTING 18 BIT NPR READ
1808	:	:	WORD COUNT INCORRECT
1809	002530	057053	EM217
1810	002532	063201	EM3020
1811	002534	050770	DT035
1812	002536	051476	DF035

1813	:	ERROR 125: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1814	:	CS1 INCORRECT
1815	002540 057106	EM218
1816	002542 061564	EM3000
1817	002544 050770	DT035
1818	002546 051476	DF035
1819	:	ERROR 126: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1820	:	CS2 INCORRECT
1821	002550 057106	EM218
1822	002552 063107	EM3018
1823	002554 050770	DT035
1824	002556 051476	DF035
1825	:	ERROR 127: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1826	:	BUS ADDRESS INCORRECT
1827	002560 057106	EM218
1828	002562 063153	EM3019
1829	002564 050770	DT035
1830	002566 051476	DF035
1831	:	ERROR 130: ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT
1832	:	WORD COUNT INCORRECT
1833	002570 057106	EM218
1834	002572 063201	EM3020
1835	002574 050770	DT035
1836	002576 051476	DF035
1837	:	ERROR 131: ATTEMPTING 18 BIT NPR READ
1838	:	DATA BUFFER INCORRECT
1839	002600 057053	EM217
1840	002602 063232	EM3021
1841	002604 051024	DT042
1842	002606 051546	DF042
1843	:	ERROR 132: ATTEMPTING 18 BIT NPR READ
1844	:	CS1 INCORRECT AFTER READING DATA BUFFER
1845	002610 057053	EM217
1846	002612 063256	EM3022
1847	002614 051052	DT055
1848	002616 051616	DF055
1849	:	ERROR 133: ATTEMPTING 18 BIT NPR READ
1850	:	CS2 INCORRECT AFTER READING DATA BUFFER
1851	002620 057053	EM217
1852	002622 063326	EM3023
1853	002624 051052	DT055
1854	002626 051616	DF055
1855	:	ERROR 134: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1856	:	CS1 INCORRECT
1857	002630 057166	EM219
1858	002632 061564	EM3000
1859	002634 050770	DT035
1860	002636 051476	DF035
1861	:	ERROR 135: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1862	:	CS2 INCORRECT
1863	002640 057166	EM219
1864	002642 063107	EM3018
1865	002644 050770	DT035
1866	002646 051476	DF035
1867	:	ERROR 136: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1868	:	BUS ADDRESS INCORRECT



1869	002650	057166	EM219
1870	002652	063153	EM3019
1871	002654	050770	DT035
1872	002656	051476	DF035
1873	:	:	ERROR 137: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1874	:	:	WORD COUNT INCORRECT
1875	002660	057166	EM219
1876	002662	063201	EM3020
1877	002664	050770	DT035
1878	002666	051476	DF035
1879	:	:	ERROR 140: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1880	:	:	CHECKING ZERO DETECT
1881	:	:	CS1 INCORRECT
1882	002670	057246	EM220
1883	002672	061564	EM3000
1884	002674	050770	DT035
1885	002676	051476	DF035
1886	:	:	ERROR 141: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1887	:	:	CHECKING ZERO DETECT
1888	:	:	CS2 INCORRECT
1889	002700	057246	EM220
1890	002702	063107	EM3018
1891	002704	050770	DT035
1892	002706	051476	DF035
1893	:	:	ERROR 142: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1894	:	:	CHECKING ZERO DETECT
1895	:	:	BUS ADDRESS INCORRECT
1896	002710	057246	EM220
1897	002712	063153	EM3019
1898	002714	050770	DT035
1899	002716	051476	DF035
1900	:	:	ERROR 143: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1901	:	:	CHECKING ZERO DETECT
1902	:	:	WORD COUNT INCORRECT
1903	002720	057246	EM220
1904	002722	063201	EM3020
1905	002724	050770	DT035
1906	002726	051476	DF035
1907	:	:	ERROR 144: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1908	:	:	DATA BUFFER INCORRECT
1909	002730	057166	EM219
1910	002732	063232	EM3021
1911	002734	051014	DT041
1912	002736	051522	DF041
1913	:	:	ERROR 145: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1914	:	:	CS1 INCORRECT AFTER READING DATA BUFFER
1915	002740	057166	EM219
1916	002742	063256	EM3022
1917	002744	051120	DT077
1918	002746	051676	DF077
1919	:	:	ERROR 146: ATTEMPTING 18 BIT NPR READ WITH BIT 16(PA) SET
1920	:	:	CS2 INCORRECT AFTER READING DATA BUFFER
1921	002750	057166	EM219
1922	002752	063326	EM3023
1923	002754	051120	DT077
1924	002756	051676	DF077

1925			:	ERRGR 147: UNEXPECTED MEMORY PARITY ENABLE TRAP
1926	002760	055340	:	EM000
1927	002762	052520	:	DH000C
1928	002764	050640	:	DT000
1929	002766	051302	:	DF000
1930			:	ERROR 150: ATTEMPTING SIMULATION OF DATA IN READ HEADER
1931			:	CS1 INCORRECT
1932	002770	057325	:	EM221
1933	002772	061564	:	EM3000
1934	002774	051134	:	DT150
1935	002776	051722	:	DF150
1936			:	ERROR 151: ATTEMPTING READ HEADER IN MAINTANENCE MODE
1937			:	CS1 INCORRECT AFTER COMPLETION OF COMMAND
1938	003000	057402	:	EM222
1939	003002	063472	:	EM3026
1940	003004	051024	:	DT042
1941	003006	051546	:	DF042
1942			:	ERROR 152: ATTEMPTING READ HEADER IN MAINTANENCE MODE
1943			:	CS2 INCORRECT AFTER COMPLETION OF COMMAND
1944	003010	057402	:	EM222
1945	003012	063541	:	EM3027
1946	003014	051024	:	DT042
1947	003016	051546	:	DF042
1948			:	ERROR 153: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
1949			:	CS1 INCORRECT AFTER UNLOADING DATA BUFFER
1950	003020	057455	:	EM223
1951	003022	063256	:	EM3022
1952	003024	051052	:	DT055
1953	003026	051616	:	DF055
1954			:	ERROR 154: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
1955			:	CS2 INCORRECT AFTER UNLOADING DATA
1956	003030	057455	:	EM223
1957	003032	063326	:	EM3023
1958	003034	051052	:	DT055
1959	003036	051616	:	DF055
1960			:	ERROR 155: ATTEMPTING DATA BUFFER READ AFTER READ HEADER
1961			:	DATA READ INCORRECT
1962	003040	057455	:	EM223
1963	003042	063232	:	EM3021
1964	003044	051040	:	DT054
1965	003046	051572	:	DF054
1966			:	ERROR 156: ATTEMPTING SIMULATION OF DATA IN READ HEADER (20 BIT FORMAT)
1967			:	CS1 INCORRECT
1968	003050	057533	:	EM224
1969	003052	061564	:	EM3000
1970	003054	051134	:	DT150
1971	003056	051722	:	DF150
1972			:	ERROR 157: ATTEMPTING READ HEADER (20 BIT FORMAT) IN MAINT MODE/
1973			:	CS1 INCORRECT AFTER COMMAND COMPLETION
1974	003060	057630	:	EM225
1975	003062	063472	:	EM3026
1976	003064	051024	:	DT042
1977	003066	051546	:	DF042
1978			:	ERROR 160: ATTEMPTING READ HEADER (20 BIT FORMAT) IN MAINT MODE
1979			:	CS2 INCORRECT AFTER COMPLETION OF COMMAND
1980	003070	057630	:	EM225

1981	003072	063541	EM3027
1982	003074	051024	DT042
1983	003076	051546	DF042
1984	:	:	ERROR 161: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
1985	:	:	CS1 INCORRECT AFTER UNLOADING DATA BUFFER
1986	003100	057715	EM226
1987	003102	063256	EM3022
1988	003104	051052	DT055
1989	003106	051616	DF055
1990	:	:	ERROR 162: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
1991	:	:	CS2 INCORRECT AFTER UNLOADING DATA BUFFER
1992	003110	057715	EM226
1993	003112	063326	EM3023
1994	003114	051052	DT055
1995	003116	051616	DF055
1996	:	:	ERROR 163: ATTEMPTING DATA BUFFER READ AFTER 20 BIT READ HEADER
1997	:	:	DATA BUFFER INCORRECT
1998	003120	057715	EM226
1999	003122	063232	EM3021
2000	003124	051040	DT054
2001	003126	051572	DF054
2002	:	:	ERROR 164: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2003	:	:	CS1 INCORRECT
2004	003130	060015	EM227
2005	003132	061564	EM3000
2006	003134	051150	DT164
2007	003136	051746	DF164
2008	:	:	ERROR 165: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2009	:	:	CS2 INCORRECT
2010	003140	060015	EM227
2011	003142	063107	EM3018
2012	003144	051150	DT164
2013	003146	051746	DF164
2014	:	:	ERROR 166: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2015	:	:	CS1 INCORRECT READING EMPTY SILO
2016	003150	060015	EM227
2017	003152	063610	EM3028
2018	003154	051150	DT164
2019	003156	051746	DF164
2020	:	:	ERROR 167: ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER
2021	:	:	CS1 INCORRECT READING EMPTY SILO
2022	003160	060015	EM227
2023	003162	063657	EM3029
2024	003164	051150	DT164
2025	003166	051746	DF164
2026	:	:	ERROR 170: WRITE BIT ERRORS
2027	003170	000000	EMW: 0
2028	003172	000000	0
2029	003174	051166	DT170
2030	003176	051772	DF170
2031	:	:	ERROR 171: WRITE GATE NOT RESET WITH SECTOR PULSE
2032	003200	060231	EM231
2033	003202	063726	EM3030
2034	003204	051212	DT171
2035	003206	052016	DF171
2036	:	:	ERROR 172: WRITE GATE NOT SET WITH SECTOR PULSE RESET

2037	003210	060353	EM232
2038	003212	063726	EM3030
2039	003214	051212	DT171
2040	003216	052016	DF171
2041	:	:	ERROR 173: WRITE GATE NOT RESET WITH SECOND INDEX PULSE
2042	003220	060601	EM235
2043	003222	063726	EM3030
2044	003224	051212	DT171
2045	003226	052016	DF171
2046	:	:	ERROR 174: CS1 INCORRECT AT END OF WRITE HEADER
2047	003230	060711	EM236
2048	003232	061564	EM3000
2049	003234	050664	DT015
2050	003236	051332	DF015
2051	:	:	ERROR 175: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2052	:	:	CS1 INCORRECT
2053	003240	061261	EM241
2054	003242	061564	EM3000
2055	003244	051222	DT175
2056	003246	052042	DF175
2057	:	:	ERROR 176: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2058	:	:	CS2 INCORRECT
2059	003250	061261	EM241
2060	003252	063107	EM3018
2061	003254	051222	DT175
2062	003256	052042	DF175
2063	:	:	ERROR 177: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2064	:	:	DRIVE STATUS REG INCORRECT
2065	003260	061261	EM241
2066	003262	063754	EM3031
2067	003264	051222	DT175
2068	003266	052042	DF175
2069	:	:	ERROR 200: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26)
2070	:	:	ERROR REG INCORRECT
2071	003270	061261	EM241
2072	003272	064007	EM3032
2073	003274	051222	DT175
2074	003276	052042	DF175
2075	:	:	ERROR 201: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2076	:	:	CS1 INCORRECT
2077	003300	061345	EM242
2078	003302	061564	EM3000
2079	003304	051222	DT175
2080	003306	052042	DF175
2081	:	:	ERROR 202: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2082	:	:	CS2 INCORRECT
2083	003310	061345	EM242
2084	003312	063107	EM3018
2085	003314	051222	DT175
2086	003316	052042	DF175
2087	:	:	ERROR 203: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2088	:	:	DRIVE STATUS REG INCORRECT
2089	003320	061345	EM242
2090	003322	063754	EM3031
2091	003324	051222	DT175
2092	003326	052042	DF175

2093	:	ERROR 204: ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24)
2094	:	ERROR REGISTER INCORRECT
2095	003330	061345
2096	003332	064007
2097	003334	051222
2098	003336	052042
2099	:	ERROR 205: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2100	:	CS1 INCORRECT
2101	003340	061431
2102	003342	061564
2103	003344	051222
2104	003346	052042
2105	:	ERROR 206: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2106	:	CS2 INCORRECT
2107	003350	061431
2108	003352	063107
2109	003354	051222
2110	003356	052042
2111	:	ERROR 207: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2112	:	DRIVE STATUS REG INCORRECT
2113	003360	061431
2114	003362	063754
2115	003364	051222
2116	003366	052042
2117	:	ERROR 210: ATTEMPTING TO CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS
2118	:	ERROR REG INCORRECT
2119	003370	061431
2120	003372	064007
2121	003374	051222
2122	003376	052042
2123	:	ERROR 211: ATTEMPTING TO CLEAR CONTROLLER ERROR
2124	:	CS1 INCORRECT
2125	003400	061532
2126	003402	061564
2127	003404	051246
2128	003406	052066
2129	:	ERROR 212: ATTEMPTING TO CLEAR CONTROLLER ERROR
2130	:	CS2 INCORRECT
2131	003410	061532
2132	003412	063107
2133	003414	051246
2134	003416	052066
2135	:	ERROR 213: ATTEMPTING TO CLEAR CONTROLLER
2136	:	DRIVE STATUS REG INCORRECT
2137	003420	061532
2138	003422	063754
2139	003424	051246
2140	003426	052066
2141	:	ERROR 214: ATTEMPTING TO CLEAR CONTROLLER
2142	:	ERROR REG INCORRECT
2143	003430	061532
2144	003432	064007
2145	003434	051246
2146	003436	052066
2147	:	TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER
2148	:	

2149	003440	000000	T.CS1:	.WORD	0	:CONTROL AND STATUS REGISTER 1
2150	003442	000000	T.WC:	.WORD	0	:WORD COUNT REGISTER
2151	003444	000000	T.BA:	.WORD	0	:BUS ADDRESS REGISTER
2152	003446	000000	T.DA:	.WORD	0	:DESIRED TRACK SECTOR REGISTER
2153	003450	000000	T.CS2:	.WORD	0	:CONTROL AND STATUS REGISTER 2
2154	003452	000000	T.DS:	.WORD	0	:DRIVE STATUS REGISTER
2155	003454	000000	T.ER:	.WORD	0	:ERROR REGISTER
2156	003456	000000	T.ASOF:	.WORD	0	:ATTENTION SUMMARY AND OFFSET REGISTER
2157	003460	000000	T.DCYL:	.WORD	0	:DESIRED CYLINDER REGISTER
2158	003462	000000	T.DB:	.WORD	0	:DATA BUFFER
2159	003464	000000	T.MR1:	.WORD	0	:MAINTENANCE REGISTER 1
2160	003466	000000	T.MR2:	.WORD	0	:MAINTENANCE REGISTER 2
2161	003470	000000	T.MR3:	.WORD	0	:MAINTENANCE REGISTER 3
2162	003472	000000	T.ECPS:	.WORD	0	:ECC POSITION INFORMATION
2163	003474	000000	T.ECPT:	.WORD	0	:ECC PATTERN INFORMATION
2164	003476	000000	T.SPARE:	.WORD	0	:SPARE REGISTER

2165  
2166 .SBTTL EXPECTED RK611 CONTROLLER REGISTERS

2168	003500	000000	E.CS1:	.WORD	0	:CONTROL AND STATUS REGISTER 1
2169	003502	000000	E.WC:	.WORD	0	:WORD COUNT REGISTER
2170	003504	000000	E.BA:	.WORD	0	:BUS ADDRESS REGISTER
2171	003506	000000	E.DA:	.WORD	0	:DESIRED TRACK SECTOR REGISTER
2172	003510	000000	E.CS2:	.WORD	0	:CONTROL AND STATUS REGISTER 2
2173	003512	000000	E.DS:	.WORD	0	:DRIVE STATUS REGISTER
2174	003514	000000	E.ER:	.WORD	0	:ERROR REGISTER
2175	003516	000000	E.ASOF:	.WORD	0	:ATTENTION SUMMARY AND OFFSET REGISTER
2176	003520	000000	E.DCYL:	.WORD	0	:DESIRED CYLINDER REGISTER
2177	003522	000000	E.DB:	.WORD	0	:DATA BUFFER
2178	003524	000000	E.MR1:	.WORD	0	:MAINTENANCE REGISTER 1
2179	003526	000000	E.MR2:	.WORD	0	:MAINTENANCE REGISTER 2
2180	003530	000000	E.MR3:	.WORD	0	:MAINTENANCE REGISTER 3
2181	003532	000000	E.ECPS:	.WORD	0	:ECC POSITION INFORMATION
2182	003534	000000	E.ECPT:	.WORD	0	:ECC PATTERN INFORMATION
2183	003536	000000	E.SPARE:	.WORD	0	:SPARE REGISTER

2184  
2185 .SBTTL PREVIOUS RK611 CONTROLLER REGISTERS

2187	003540	000000	P.CS1:	.WORD	0	:CONTROL AND STATUS REGISTER 1
2188	003542	000000	P.WC:	.WORD	0	:WORD COUNT REGISTER
2189	003544	000000	P.BA:	.WORD	0	:BUS ADDRESS REGISTER
2190	003546	000000	P.DA:	.WORD	0	:DESIRED TRACK SECTOR REGISTER
2191	003550	000000	P.CS2:	.WORD	0	:CONTROL AND STATUS REGISTER 2
2192	003552	000000	P.DS:	.WORD	0	:DRIVE STATUS REGISTER
2193	003554	000000	P.ER:	.WORD	0	:ERROR REGISTER
2194	003556	000000	P.ASOF:	.WORD	0	:ATTENTION SUMMARY AND OFFSET REGISTER
2195	003560	000000	P.DCYL:	.WORD	0	:DESIRED CYLINDER REGISTER
2196	003562	000000	P.DB:	.WORD	0	:DATA BUFFER
2197	003564	000000	P.MR1:	.WORD	0	:MAINTENANCE REGISTER 1
2198	003566	000000	P.MR2:	.WORD	0	:MAINTENANCE REGISTER 2
2199	003570	000000	P.MR3:	.WORD	0	:MAINTENANCE REGISTER 3
2200	003572	000000	P.ECPS:	.WORD	0	:ECC POSITION INFORMATION
2201	003574	000000	P.ECPT:	.WORD	0	:ECC PATTERN INFORMATION
2202	003576	000000	P.SPARE:	.WORD	0	:SPARE REGISTER

2203 .SBTTL PROGRAM DEFINED VARIABLES  
2204

```
2205 003600 000210      RKVEC: .WORD 210      ;RK611 VECTOR
2206 003602 000240      RKPRI: .WORD PR5     ;RK611 PRIORITY
2207 003604 000000      TRAPPC: .WORD 0     ;PC FOR MEMORY CHECK ENABLE TRAP
2208 003606 000000      SRTFLG: .WORD 0     ;START FLAG
2209                      ; 0 = 200
2210                      ; 1 = 214
2211                      ; -1 = 204
2212 003610 000000      ERRCNT: .WORD 0     ;ERROR COUNT FOR SWITCH 12 ABORT
2213 003612 000000      P1.BIT: .WORD 0     ;NEXT BIT IN DATA SIMULATION
2214 003614 000000      PR.BIT: .WORD 0     ;PRESENT BIT IN DATA SIMULATION
2215 003616 000000      M1.BIT: .WORD 0     ;PREVIOUS BIT IN DATA SIMULATION
2216 003620 000000      M2.BIT: .WORD 0     ;BIT BEFORE PREVIOUS BIT
2217 003622 000000      BITCNT: .WORD 0     ;BIT POSITION
2218 003624 000000      WRDCNT: .WORD 0     ;WORD COUNT FOR NPR TRANSFER
2219 003626 000000      SECCNT: .WORD 0     ;SECTOR COUNT
2220 003630 000000      MEMPAR: .WORD 0     ;MEMORY EMABLE ON FIRST 24K
2221 003632 000015      WAITIM: .WORD 15    ;WAIT TIME FOR CONTROLLER READY
2222 003634 000000      SAVSWR: .WORD 0     ;STORAGE FOR SWITCH REGISTER
2223 003636 000000      PARPRE: .WORD 0     ;PARITY OPTION PRESENT FOR
2224                      ; LOCATION USED
2225 003640 000000      CSRPTR: .WORD 0     ;POINTER TO CSR TO BE USED
2226                      .SBTTL PROGRAM SETUP
2227
2228 003642 012737 000001 003606 PARM:  MOV #1,SRTFLG  ;LOAD START FLAG FOR PARMETER START
2229 003650 000406                      BR          START1
2230
2231 003652 012737 177777 003606 PESTRT: MOV #-1,SRTFLG ;LOAD START FLAG FOR RESTART
2232 003660 000402                      BR          START1
2233
2234 003662 005037 003606      START: CLR SRTFLG      ;CLEAR START FLAG
2235 003666 000005      START1: RESET      ;RESET THE WHOLE SYSTEM
2236 003670 012706 001100      MOV #STACK,SP      ;INITIALIZE STACK POINTER
2237 003674 004737 046724      JSR PC,$KINT      ;INIT KEYBOARD
2238 003700 012746 000340      MOV #PR7,-(SP)     ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
2239 003704 012746 003712      MOV #1$,-(SP)     ;LOAD START OF PROGRAM
2240 003710 000002      RTI              ;LOAD PSW
2241
2242 003712
2243
2244
2245 003712 012706 001100      1$:
2246 003716 005026                      .SBTTL INITIALIZE THE COMMON TAGS
2247 003720 022706 001140      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2248 003724 001374                      MOV #CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
2249 003726 012706 001100      CLR (R6)+          ;;CLEAR MEMORY LOCATION
2250                      CMP #SWR,R6 ;;DONE?
2251                      BNE .-6 ;;LOOP BACK IF NO
2252                      MOV #STACK,SP ;;SETUP THE STACK POINTER
2253                      ;;INITIALIZE A FEW VECTORS
2254 003732 012737 044220 000020      MOV #SCOPE,@IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
2255 003740 012737 000340 000022      MOV #340,@IOTVEC+2 ;;LEVEL 7
2256 003746 012737 045146 000030      MOV #ERROR,@EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
2257 003754 012737 000340 000032      MOV #340,@EMTVEC+2 ;;LEVEL 7
2258 003762 012737 050550 000034      MOV #STRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2259 003770 012737 000340 000036      MOV #340,@TRAPVEC+2;LEVEL 7
2260 003776 012737 050420 000024      MOV #SPWRDN,@PWRVEC ;;POWER FAILURE VECTOR
2261 004004 012737 000340 000026      MOV #340,@PWRVEC+2 ;;LEVEL 7
2262 004012 013737 042414 042406      MOV SENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
2263 004020 005037 001200      CLR $TIMES        ;;INITIALIZE NUMBER OF ITERATIONS
```

```

2261 004024 005037 001202          CLR    $ESCAPE          ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2262 004030 112737 000001 001115  MOV    #1,$ERMAX        ;;ALLOW ONE ERROR PER TEST
2263 004036 012737 004036 001106  MOV    #,$SLPADR        ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2264 004044 012737 004044 001110  MOV    #,$SLPERR        ;;SETUP THE ERROR LOOP ADDRESS
2265                                     ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2266                                     ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2267 004052 013746 000004          MOV    @#ERRVEC,-(SP)    ;;SAVE ERROR VECTOR
2268 004056 012737 004112 000004  MOV    #64$,@#ERRVEC    ;;SET UP ERROR VECTOR
2269 004064 012737 177570 001140  MOV    #DSWR,SWR        ;;SETUP FOR A HARDWARE SWICH REGISTER
2270 004072 012737 177570 001142  MOV    #DDISP,DISPLAY    ;;AND A HARDWARE DISPLAY REGISTER
2271 004100 022777 177777 175037  CMP    #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
2272 004106 001012          BNE    66$              ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2273                                     ;;AND THE HARDWARE SWR IS NOT -1
2274 004110 000403          BR     65$              ;;BRANCH IF NO TIMEOUT
2275 004112 012716 004120          64$: MOV    #65$,(SP)      ;;SET UP FOR TRAP RETURN
2276 004116 000002          RTI
2277 004120 012737 000176 001140  65$: MOV    #SWREG,SWR    ;;POINT TO SOFTWARE SWR
2278 004126 012737 000174 001142  MOV    #DISPREG,DISPLAY
2279 004134 012637 000004          66$: MOV    (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
2280
2281 004140 005037 001222          CLR    $PASS           ;;CLEAR PASS COUNT
2282 004144 132737 000200 001235  BITB   #APTSIZE,$ENVM   ;;TEST USER SIZE UNDER APT
2283 004152 001403          BEQ    67$             ;;YES,USE NON-APT SWITCH
2284 004154 012737 001236 001140  MOV    #$$SWREG,SWR    ;;NO,USE APT SWITCH REGISTER
2285 004162          67$:
2286 004162 005037 003610          CLR    ERRCNT          ;;CLEAR ERROR COUNT FOR SWITCH 12 ABORT
2287 .SBTTL TYPE PROGRAM NAME
2288 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2289 004166 005227 177777          INC    #-1             ;;FIRST TIME?
2290 004172 001047          BNE    68$             ;;BRANCH IF NO
2291 004174 022737 042550 000042  CMP    #SENDAD,@#42    ;;ACT-11?
2292 004202 001443          BEQ    68$             ;;BRANCH IF YES
2293 004204 104401 004252          TYPE   ,69$           ;;TYPE ASCIZ STRING
2294 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2295 004210 005737 000042          TST   @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
2296 004214 001012          BNE    70$             ;;BRANCH IF YES
2297 004216 123727 001234 000001  CMPB   $ENV,#1         ;;ARE WE RUNNING UNDER APT?
2298 004224 001406          BEQ    70$             ;;BRANCH IF YES
2299 004226 023727 001140 000176  CMP    SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
2300 004234 001005          BNE    71$             ;;BRANCH IF NO
2301 004236 104406          GTSWR                    ;;GET SOFT-SWR SETTINGS
2302 004240 000403          BR     71$
2303 004242 112737 000001 001134  70$: MOV    #1,$AUTOB     ;;SET AUTO-MODE INDICATOR
2304 004250          71$:
2305 004250 000420          BR     68$             ;;GET OVER THE ASCIZ
2306 ;;69$: .ASCIZ <CRLF>/CZR6CEO RK611 DSKLS CTRL PRT3/<CRLF>
2307 68$:
2308 004312 005227 177777          INC    #-1             ;;TEST IF FIRST PASS
2309 004316 001002          BNE    6$              ;;NO - SKIP
2310 004320 104401 052234          TYPE   ,OPR006        ;;TYPE RUN TIME MESSAGE
2311 004324 022737 000001 003606  6$: CMP    #1,SRTFLG    ;;CHECK IF PARAMETER START
2312 004332 001122          BNE    15$            ;;NO, CONTINUE SETUP
2313 004334 104401 052122          5$: TYPE   ,OPR001    ;;TYPE 'RK611 BUS ADDRESS ( ) -'
2314 004340 013746 001270          MOV    $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
2315 004344 104402          TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2316 004346 104401 052154          TYPE   ,OPR002

```



```
2317 004352 104412 RDOCT ;GET VALUE
2318 004354 012637 001160 MOV (SP)+,$TMP0
2319 004360 001407 BEQ 7$ ;CHECK IF <CR>
2320 004362 022737 160000 001160 CMP #160000,$TMP0 ;CHECK IF IN I/O PAGE
2321 004370 101361 BHI 5$
2322 004372 013737 001160 001270 MOV $TMP0,$BASE ;LOAD NEW BUS ADDRESS
2323 004400 104401 052162 7$: TYPE ,OPR003 ;TYPE 'RK611 VECTOR ADDRESS ( ) -'
2324 004404 013746 001264 MOV $VECT1,-(SP) ;TYPE OUT VECTOR ADDRESS
2325 004410 042716 160000 BIC #160000,(SP)
2326 004414 104402 TYPOC
2327 004416 104401 052154 TYPE ,OPR002
2328 004422 104412 RDOCT ;GET VALUE
2329 004424 012637 001160 MOV (SP)+,$TMP0
2330 004430 001412 BEQ 10$ ;CHECK IF <CR>
2331 004432 022737 001000 001160 CMP #1000,$TMP0 ;CHECK IF LEGAL
2332 004440 101757 BLOS 7$
2333 004442 042737 017777 001264 BIC #17777,$VECT1 ;LOAD NEW VECTOR ADDRESS
2334 004450 053737 001160 001264 BIS $TMP0,$VECT1
2335 004456 104401 052212 10$: TYPE ,OPR004 ;TYPE 'RK611 PRIORITY ( ) -'
2336 004462 005046 CLR -(SP)
2337 004464 113716 001265 MOVB $VECT1+1,(SP)
2338 004470 006216 ASR (SP) ;SHIFT 5 BITS RIGHT
2339 004472 006216 ASR (SP)
2340 004474 006216 ASR (SP)
2341 004476 006216 ASR (SP)
2342 004500 006216 ASR (SP)
2343 004502 104402 TYPOC
2344 004504 104401 052154 TYPE ,OPR002
2345 004510 104412 RDOCT ;GET VALUE
2346 004512 012637 001160 MOV (SP)+,$TMP0
2347 004516 001430 BEQ 15$ ;CHECK FOR DEFAULT
2348 004520 022737 000007 001160 CMP #7,$TMP0 ;CHECK IF LEGAL
2349 004526 103753 BLO 10$
2350 004530 022737 000004 001160 CMP #4,$TMP0
2351 004536 101347 BHI 10$
2352 004540 006337 001160 ASL $TMP0 ;SHIFT 5 BITS LEFT
2353 004544 006337 001160 ASL $TMP0
2354 004550 006337 001160 ASL $TMP0
2355 004554 006337 001160 ASL $TMP0
2356 004560 006337 001160 ASL $TMP0
2357 004564 042737 160000 001264 BIC #160000,$VECT1 ;STORE NEW PRIORITY
2358 004572 153737 001160 001265 BISB $TMP0,$VECT1+1
2359 004600 013737 001264 003600 15$: MOV $VECT1,RKVEC ;STORE RK611 VECTOR
2360 004606 042737 160000 003600 BIC #160000,RKVEC
2361 004614 113737 001265 003602 MOVB $VECT1+1,RKPRI ;STORE PRIORITY
2362 004622 004737 043712 JSR PC,$SIZE ;SIZE MEMORY
2363 004626 013702 001270 MOV $BASE,R2 ;SET RK611 BASE
2364 004632 005037 001202 CLR $ESCAPE ;CLEAR ESCAPE
2365
2366 004636 012746 00000C NEWPAS: MOV #PRO,-(SP) ;ALLOW ALL INTERRUPTS
2367 004642 012746 00465C MOV #TST1,-(SP)
2368 004646 000002 RTI
```

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  
2424

.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  
MOV #3\*4+2,R0 ;CLOCK IN DRIVE MESSAGE  
1\$: MOV #DMD.MCLK,RKMR1(R2)  
MOV #DMD,RKMR1(R2)  
DFC 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.  
\*

```
2425          :*          CYLINDER 1777, HEAD 7, DRIVE 7.
2426          :*
2427          :*-----*
2428 005066 000004 TST2: SCOPE
2429 005070 012737 000144 001200 MOV #100.,$TIMES ;;DO 100. ITERATIONS
2430 005076 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2431 005102 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2432 005110 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2433 005116 012762 001777 000020 MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2434 005124 012762 003400 000006 MOV #3400,RKDA(R2) ;LOAD TRACK
2435 005132 012762 000007 000010 MOV #7,RKCS2(R2) ;LOAD DRIVE NUM.
2436 005140 012762 012027 000000 MOV #CDT!CFMT!WRHEAD,RKCS1(R2) ;ISSUE WRHEAD WITH CDT SET IN
2437          : 24 SECTOR FORMAT
2438 005146 012700 000016 MOV #3*4+2,R0 ;CLOCK IN DRIVE MESSAGE
2439 005152 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
2440 005160 012762 000040 000026 MOV #DMD,RKMR1(R2)
2441 005166 005300 DEC R0
2442 005170 001370 BNE 1$
2443 005172 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND STATUS REG. 1
2444 005200 016237 000034 003466 MOV RKMR2(R2),T.MR2 ;STORE MAINT REG. 2 (MESS A)
2445 005206 016237 000036 003470 MOV RKMR3(R2),T.MR3 ;STORE MAINT REG. 3 (MESS B)
2446 005214 012737 012027 003500 MOV #CDT!CFMT!WRHEAD,E.CS1 ;LOAD EXPECTED CS1
2447 005222 012737 071227 003526 MOV #S.SEEK!S.RTC!S.FMT!70007,E.MR2 ;LOAD EXPECTED MR2
2448 005230 012737 037760 003530 MOV #37760,E.MR3 ;LOAD EXPECTED MR3
2449 005236 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2450 005244 001405 BEQ 2$ ;YES, CHECK MESSAGE A
2451 005246 104004 ERROR 4 ;CS1 INCORRECT
2452 005250 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2453 005256 000412 BR TST3 ;GO TO NEXT TEST
2454 005260 023737 003526 003466 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2455 005266 001401 BEQ 3$ ;YES, CHECK MESSAGE B
2456 005270 104005 ERROR 5 ;MESS A INCORRECT
2457 005272 023737 003530 003470 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2458 005300 001401 BEQ TST3 ;YES, GO ON TO NEXT TEST
2459 005302 104006 ERROR 6 ;MESS B INCORRECT
```

```
2460
2461 :*-----*
2462 :*TEST 3 READ HEADER DRIVE CLEAR MESSAGE
2463 :*
2464 :* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
2465 :* IN DIAGNOSTIC MODE. ISSUE A READ HEADER WITH CDT SET
2466 :* IN 24 SECTOR FORMAT, CYLINDER 1777, HEAD 7, DRIVE 7.
2467 :* CLOCK SEEK MESSAGE AND MAKE SURE A DRIVE CLEAR IS
2468 :* GENERATED AND THE PROPER BITS ARE SET.
2469 :*
2470 :*-----*
```

```
2471 005304 000004 TST3: SCOPE
2472 005306 012737 000144 001200 MOV #100.,$TIMES ;;DO 100. ITERATIONS
2473 005314 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2474 005320 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2475 005326 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2476 005334 012762 001777 000020 MOV #1777,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG.
2477 005342 012762 003400 000006 MOV #3400,RKDA(R2) ;LOAD TRACK
2478 005350 012762 000007 000010 MOV #7,RKCS2(R2) ;LOAD DRIVE NUMBER
2479 005356 012762 012025 000000 MOV #CDT!CFMT!RDHEAD,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2480          : 24 SECTOR FORMAT
```

```

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

```

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

```

2515 005520 000004      TST4:  SCOPE
2516 005522 012737 000144 001200  MOV      #100.,$TIMES    ;DO 100. ITERATIONS
2517 005530 013702 001270      MOV      $BASE,R2       ;LOAD RK611 BASE
2518 005534 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2519 005542 012762 000040 000026  MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2520 005550 012762 001777 000020  MOV      #1777,RKDC.L(R2) ;LOAD CYLINDER ADDRESS REG.
2521 005556 012762 003400 000006  MOV      #3400,RKDA(R2) ;LOAD TRACK
2522 005564 012762 000007 000010  MOV      #7,RKCS2(R2)   ;LOAD DRIVE NUMBER
2523 005572 012762 012027 000000  MOV      #CDT!CFMT!WRHEAD,RKCS1(R2) ;ISSUE COMMAND WITH CDT SET IN
2524      ;      24 SECTOR FORMAT
2525 005600 012700 000156      MOV      #27.*4+2,R0    ;LOAD COUNT TO LOAD DRIVE CLEAR
2526 005604 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2)
2527 005612 012762 000040 000026  MOV      #DMD,RKMR1(R2)
2528 005620 005300      DEC      R0
2529 005622 001370      BNE     1$
2530 005624 016237 000000 003440  MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG.
2531 005632 016237 000034 003466  MOV      RKMR2(R2),T.MR2 ;STORE MAINT. REG. 2 (MESS A)
2532 005640 016237 000036 003470  MOV      RKMR3(R2),T.MR3 ;STORE MAINT. REG. 3 (MESS B)
2533 005646 012737 012027 003500  MOV      #CDT!CFMT!WRHEAD,E.CS1 ;LOAD EXPECTED CS1
2534 005654 012737 071407 003526  MOV      #S.CLR!S.FMT!70007,E.MR2 ;LOAD EXPECTED MAINT REG. 2
2535 005662 005037 003530      CLR     E.MR3           ;LOAD EXPECTED MAINT REG.
2536 005666 023737 003500 003440  CMP      E.CS1,T.CS1    ;CHECK COMMAND AND STATUS REG 1 CORRECT
  
```

```

2537 005674 001405 BEQ 2$ ;YES, CHECK CS2
2538 005676 104012 ERROR 12 ;CS1 INCORRECT
2539 005700 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2540 005706 000412 BR TST5 ;GO TO NEXT TEST
2541 005710 023737 003526 003466 2$: CMP E.MR2,T.MR2 ;CHECK MESS A CORRECT
2542 005716 001401 BEQ 3$ ;YES, CHECK MESS B
2543 005720 104013 ERROR 13 ;MESS A INCORRECT
2544 005722 023737 003530 003470 3$: CMP E.MR3,T.MR3 ;CHECK MESS B CORRECT
2545 005730 001401 BEQ TST5 ;YES, GO ON TO NEXT TEST
2546 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.
  
```

```

*****
2563 005734 000004 *ST5: SCOPE
2564 005736 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS
2565 005744 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2566 005750 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2567 005756 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2568 005764 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;INITIALIZE ROM ADDRESS
2569 005772 012762 000040 000026 MOV #DMD,RKMR1(R2)
2570 006000 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
2571 006006 012700 000312 MOV #50.*4+2,R0 ;CLOCK UNTIL READY FOR SECTOR PULSE
2572 006012 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
2573 006020 012762 000040 000026 MOV #DMD,RKMR1(R2)
2574 006026 005300 DEC R0
2575 006030 001370 BNE 1$
2576 006032 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2577 006040 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
2578 006046 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2579 006054 001405 BEQ 2$ ;YES, CLOCK ZEROES
2580 006056 104015 ERROR 15 ;CS1 INCORRECT
2581 006060 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2582 006066 000553 BR TST6 ;GO ON TO NEXT TEST
2583
2584 006070 012762 000140 000026 2$: MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
2585 006076 012762 000040 000026 MOV #DMD,RKMR1(R2)
2586 006104 012737 022040 003624 MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECT MAINT REG. 1
2587 006112 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
2588 006116 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT AND PREVIOUS
2589 006122 005037 003616 CLR M1.BIT ;BITS TO GENERATE ZEROES
2590 006126 012700 000200 MOV #128,R0 ;GENERATE 128 ZEROS UNTIL READ GATE
2591 006132 004737 043540 5$: JSR PC,RDBIT ;READ A ZERO
2592 006136 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINT REG. 1
  
```

```

2593 006144 023737 003524 003464      CMP      E.MR1,T.MR1      ;CHECK MAINTENANCE REG. 1 CORRECT
2594 006152 001405                      BEQ      6$              ;YES, SIMULATE NEXT BIT
2595 006154 104017                      ERROR   17              ;MAINT REG. 1 INCORRECT
2596 006156 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2597 006164 000514                      BR       TST6           ;;GO ON TO NEXT TEST
2598
2599 006166 005237 003622      6$:     INC      BITCNT      ;INCREMENT BIT COUNT
2600 006172 005300                      DEC      R0              ;CHECK READY OF READ GATE
2601 006174 001356                      BNE     5$              ;NO, CONTINUE
2602 006176 012737 122040 003524      MOV      #DMD!MEWD!ECCW!RDGATE,E.MR1 ;LOAD EXPECTED MR1
2603 006204 012700 000177                      MOV      #127,R0        ;GENERATE 127 ZEROS
2604 006210 004737 043540      10$:   JSR      PC,RDBIT      ;READ A ZERO
2605 006214 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MAINT REG. 1
2606 006222 023737 003524 003464      CMP      E.MR1,T.MR1      ;CHECK MAINT REG. 1 CORRECT
2607 006230 001405                      BEQ      11$           ;YES, SIMULATE NEXT BIT
2608 006232 104017                      ERROR   17              ;MAINT REG. 1 INCORRECT
2609 006234 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2610 006242 000465                      BR       TST6           ;;GO ON TO NEXT TEST
2611
2612 006244 005237 003622      11$:   INC      BITCNT      ;INCREMENT BIT COUNT
2613 006250 005300                      DFC     R0              ;CHECK IF ALL ZEROS ISSUED
2614 006252 001356                      BNE     10$            ;NO, CONTINUE
2615 006254 012737 000001 003614      MOV      #1,PR.BIT      ;LOAD ONE FOR READING 1
2616 006262 004737 043540      JSR      PC,RDBIT      ;READ A ONE
2617 006266 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG.
2618 006274 023737 003524 003464      CMP      E.MR1,T.MR1      ;CHECK MAINT REG. 1
2619 006302 001405                      BEQ      12$           ;YES, CONTINUE
2620 006304 104017                      ERROR   17              ;MAINTENANCE REG. 1 INCORRECT
2621 006306 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2622 006314 000440                      BR       TST6           ;;GO ON TO NEXT TEST
2623 006316 005237 003622      12$:   INC      BITCNT      ;INCREMENT BIT COUNT
2624 006322 013737 003614 003616      MOV      PR.BIT,M1.BIT  ;LOAD ZERO FOR NEXT BIT
2625 006330 005037 003614                      CLR     PR.BIT
2626 006334 004737 043540      JSR      PC,RDBIT      ;SIMULATE ZERO
2627 006340 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2628 006346 023737 003524 003464      CMP      E.MR1,T.MR1      ;CHECK MAINT REG. 1 CORRECT
2629 006354 001405                      BEQ      13$           ;CHECK CS1 CORRECT
2630 006356 104017                      ERROR   17              ;MAINTENANCE REG. 1 INCORRECT
2631 006360 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2632 006366 000413                      BR       TST6           ;;GO TO NEXT TEST
2633
2634 006370 016237 000000 003440 13$:   MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2635 006376 012737 000025 003500      MOV      #RDHEAD,E.CS1  ;LOAD EXPECTED CS1
2636 006404 023737 003500 003440      CMP      E.CS1,T.CS1      ;CHECK CS1 CORRECT
2637 006412 001401                      BEQ      TST6           ;;YES, GO TO NEXT TEST
2638 006414 104016                      ERROR   16              ;CS1 INCORRECT
2639

```

```

2640 *****
2641 *TEST 6          SECTOR PULSE DETECT IN READ HEADER (PART 2)
2642 *
2643 *
2644 * CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
2645 * IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
2646 * IN 26 SECTOR MODE, CYLINDER 0, HEAD 0, DRIVE 0.
2647 * CLOCK BOTH SEEK ANY DRIVE CLEAR MESSAGES.
2648 * SIMULATE INDEX PULSE, 255 ZEROES AND A ONE.

```

```

2649
2650
2651
2652 006416 000004
2653 006420 012737 000012 001200
2654 006426 013702 001270
2655 006432 012762 100000 000000
2656 006440 012762 000040 000000
2657 006446 012762 000140 000026
2658 006454 012762 000040 000026
2659 006462 012762 000025 000000
2660 006470 012700 000312
2661 006474 012762 000440 000026 1$:
2662 006502 0 2762 000040 000026
2663 006510 005300
2664 006512 001370
2665 006514 016237 000000 003440
2666 006522 012737 000025 003500
2667 006530 023737 003500 003440
2668 006536 001405
2669 006540 104015
2670 006542 012762 100000 000000
2671 006550 000535
2672 006552
2673 006552 012762 000240 000026 2$:
2674 006560 012700 000004
2675 006564 012762 000640 000026 3$:
2676 006572 012762 000240 000026
2677 006600 005300
2678 006602 001370
2679 006604 012762 000040 000026
2680 006612 005037 003622
2681 006616 012737 022040 003524
2682 006624 005037 003614
2683 006630 005037 003616
2684 006634 012700 000377
2685 006640 004737 043540 5$:
2686 006644 016237 000026 003464
2687 006652 023737 003524 003464
2688 006660 001405
2689 006662 104020
2690 006664 012762 100000 000000
2691 006672 000464
2692
2693 006674 005237 003622 6$:
2694 006700 005300
2695 006702 001356
2696 006704 012737 000001 003614
2697 006712 004737 043540
2698 006716 016237 000026 003464
2699 006724 023737 003524 003464
2700 006732 001405
2701 006734 104020
2702 006736 012762 100000 000000
2703 006744 000437
2704

```

```

: * MAKE SURE READ GATE DOES NOT SET.
: *
: *****
TST6: SCOPE
MOV #10.,$TIMES ;;DO 10. ITERATIONS
MOV $BASE,R2 ;;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKCS1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #DMD!MSP,RKMR1(R2) ;INITIALIZE ROM ADDRESS
MOV #DMD,RKMR1(R2)
MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
MOV #50.*4+2,R0 ;CLOCK UNTIL READY FOR SECTOR PULSE
1$: MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
BEQ 2$ ;YES, CLOCK IN ZEROS
ERROR 1$
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
BR TST7 ;;GO ON TO NEXT TEST
2$: MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDX PULSE
MOV #4,R0
3$: MOV #DMD.MIND.MCLK,RKMR1(R2)
MOV #DMD.MIND,RKMR1(R2)
DEC R0
BNE 3$
MOV #DMD,RKMR1(R2)
CLR BITCNT ;INITIALIZE BIT COUNT
MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MAINTENANCE REG. 1
CLR PR.BIT ;INITIALIZE PRESENT AND PREVIOUS
CLR M1.BIT ;BITS TO GENERATE ZEROS
MOV #255.,R0 ;GENERATE 255 ZEROES
5$: JSR PC,RDBIT ;READ A ZERO
MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
CMP E.MR1,T.MR1 ;CHECK READ GATE NOT SET
BEQ 6$ ;READ GATE NOT SET SIMULATE NEXT BIT
ERROR 2$ ;MAINT REG. 1 INCORRECT
MOV #CCLR,RKCS1(R2) ;ISSUE CONTROLLER CLEAR
BR TST7 ;;GO ON TO NEXT TEST
6$: INC BITCNT ;INCREMENT BIT COUNT
DEC R0 ;CHECK IF ALL ZEROES ISSUED
BNE 5$ ;NO, CONTINUE
MOV #1,PR.BIT ;LOAD ONE FOR READING 1
JSR PC,RDBIT ;READ A ONE
MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
CMP E.MR1,T.MR1 ;CHECK READ GATE NOT SET
BEQ 7$ ;YES, CONTINUE
ERROR 2$ ;MAINT REG. 1 INCORRECT
MOV #CCLR,RKCS1(R2) ;ISSUE CONTROLLER CLEAR
BR TST7 ;;GO ON TO NEXT TEST

```



```
2705 006746 005237 003622 7$: INC BITCNT ;INCREMENT BIT COUNT
2706 006752 013737 003614 003616 MOV PR.BIT,M1.BIT ;LOAD ZERO FOR NEXT BIT
2707 006760 005037 003614 CLR PR.BIT
2708 006764 004737 043540 JSR PC,RDBIT ;SIMULATE ZERO
2709 006770 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2710 006776 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MAINTENANCE REG. 1 CORRECT
2711 007004 001404 BEQ 9$ ;CHECK CS1 CORRECT
2712 007006 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2713 007014 000413 BR TST7 ;GO TO NEXT TEST
2714
2715 007016 016237 000000 003440 9$: MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2716 007024 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
2717 007032 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
2718 007040 001401 BEQ TST7 ;YES, GO TO NEXT TEST
2719 007042 104016 ERROR 16 ;CS1 INCORRECT
```

```
*****
*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 DOFS NOT SET.
*****
```

```
2732
2733 007044 000004 TST7 SCOPE
2734 007046 012737 000012 001200 MOV #10,$TIMES ;DO 10. ITERATIONS
2735 007054 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
2736 007060 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2737 007066 012762 000040 000000 MOV #DMD,RKCS1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2738 007074 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;INITIALIZE ROM ADDRESS
2739 007102 012762 000040 000026 MOV #DMD,RKMR1(R2)
2740 007110 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
2741 007116 012700 000312 MOV #50.*4+2,R0 ;CLOCK UNTIL READY FOR SECTOR PULSE
2742 007122 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
2743 007130 012762 000040 000026 MOV #DMD,RKMR1(R2)
2744 007136 005300 DEC R0
2745 007140 001370 BNE 1$
2746 007142 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2747 007150 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
2748 007156 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2749 007164 001405 BEQ 2$ ;YES, CLOCK IN ZEROS
2750 007166 104015 ERROR 15
2751 007170 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
2752 007176 000515 BR TST10 ;GO ON TO NEXT TEST
2753
2754 007200 005037 003622 2$: CLR BITCNT ;INITIALIZE BIT COUNT
2755 007204 012737 022040 003524 MOV #DMD!MEWD!ECCW,E.MR1 ;LOAD EXPECTED MAINTENANCE REG. 1
2756 007212 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT AND PREVIOUS
2757 007216 005037 003616 CLR M1.BIT ;BITS TO GENERATE ZEROS
2758 007222 012700 000377 MOV #255,R0 ;GENERATE 255 ZEROES
2759 007226 004737 043540 JSR PC,RDBIT ;READ A ZERO
2760 007232 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;STORE MAINTENANCE REG. 1
```



```

2761 007240 023737 003524 003464      CMP      E.MR1,T.MR1      ;CHECK READ GATE NOT SET
2762 007246 001405                      BEQ      6$              ;READ GATE NOT SET SIMULATE NEXT BIT
2763 007250 104021                      ERROR    21              ;MAINT REG. 1 INCORRECT
2764 007252 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;ISSUE CONTROLLER CLEAR
2765 007260 000464                      BR       TST10           ;;GO ON TO NEXT TEST
2766
2767 007262 005237 003622      6$:      INC      BITCNT        ;INCREMENT BIT COUNT
2768 007266 005300                      DEC      R0              ;CHECK IF ALL ZEROES ISSUED
2769 007270 001356                      BNE     5$              ;NO, CONTINUE
2770 007272 012737 000001 003614      MOV      #1,PR.BIT      ;LOAD ONE FOR READING 1
2771 007300 004737 043540      JSR     PC,RDBIT        ;READ A ONE
2772 007304 016237 000026 003464      MOV      RKMRI(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2773 007312 023737 003524 003464      CMP      E.MR1,T.MR1    ;CHECK READ GATE NOT SET
2774 007320 001405                      BEQ      7$              ;YES, CONTINUE
2775 007322 104021                      ERROR    21              ;MAINT REG. 1 INCORRECT
2776 007324 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;ISSUE CONTROLLER CLEAR
2777 007332 000437                      BR       TST10           ;;GO ON TO NEXT TEST
2778
2779 007334 005237 003622      7$:      INC      BITCNT        ;INCREMENT BIT COUNT
2780 007340 013737 003614 003616      MOV      PR.BIT,M1.BIT  ;LOAD ZERO FOR NEXT BIT
2781 007346 005037 003614                      CLR     PR.BIT
2782 007352 004737 043540      JSR     PC,RDBIT        ;SIMULATE ZERO
2783 007356 016237 000026 003464      MOV      RKMRI(R2),T.MR1 ;STORE MAINTENANCE REG. 1
2784 007364 023737 003524 003464      CMP      E.MR1,T.MR1    ;CHECK MAINTENANCE REG. 1 CORRECT
2785 007372 001404                      BEQ      9$              ;CHECK CS1 CORRECT
2786 007374 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2787 007402 000413                      BR       TST10           ;;GO TO NEXT TEST
2788
2789 007404 016237 000000 003440      9$:      MOV      RKCS1(R2),T.CS1  ;STORE COMMAND AND STATUS REG. 1
2790 007412 012737 000025 003500      MOV      #RDHEAD,E.CS1  ;LOAD EXPECTED CS1
2791 007420 023737 003500 003440      CMP      E.CS1,T.CS1    ;CHECK CS1 CORRECT
2792 007426 001401                      BEQ      TST10           ;;YES, GO TO NEXT TEST
2793 007430 104016                      ERROR    16              ;CS1 INCORRECT
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809 007432 000004                      TST10: SCOPE
2810 007434 012737 000012 001200      MOV      #10,$TIMES     ;;DO 10. ITERATIONS
2811 007442 013702 001270                      MOV      $BASE,R2       ;LOAD RK611 BASE
2812 007446 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
2813 007454 012762 000040 000026      MOV      #DMD,RKMRI(R2) ;PUT RK611 IN DIAGNOSTIC MODE
2814 007462 012762 000140 000026      MOV      #DMD!MSP,RKMRI(R2) ;INITIALIZE ROM ADDRESS
2815 007470 012762 000040 000026      MOV      #DMD,RKMRI(R2)
2816 007476 012762 064714 000004      MOV      #WRBUFF,RKBA(R2) ;LOAD BUS ADDRESS
  
```

```

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

```

2817 007504 012762 177777 000002      MOV      #-1,RKWC(R2)      ;LOAD WORD COUNT
2818 007512 012762 000027 000000      MOV      #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
2819 007520 012700 000312      MOV      #50.*4+2,R0      ;CLOCK UNTIL READY FOR INDEX PULSE
2820 007524 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2)
2821 007532 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2822 007540 005300      DEC      R0
2823 007542 001370      BNE      1$
2824 007544 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
2825 007552 016237 000004 003444      MOV      RKBA(R2),T.BA    ;STORE BUS ADDRESS REG.
2826 007560 016237 000002 003442      MOV      RKWC(R2),T.WC    ;STORE WORD COUNT REG.
2827 007566 012737 000027 003500      MOV      #WRHEAD,E.CS1   ;LOAD EXPECTED CS1
2828 007574 012737 064714 003504      MOV      #WRBUFF,E.BA    ;LOAD EXPECTED BUS ADDRESS
2829 007602 012737 177777 003502      MOV      #-1,E.WC        ;LOAD EXPECTED WORD COUNT
2830 007610 023737 003500 003440      CMP      E.CS1,T.CS1     ;CHECK COMMAND AND STATUS REG. 1 CORRECT
2831 007616 001403      BEQ      2$              ;YES, CONTINUE
2832 007620 104022      ERROR   22              ;CS1 INCORRECT
2833 007622 000137 010550      JMP      60$            ;CLEAR CONTROLLER
2834
2835 007626 023737 003504 003444 2$:      CMP      E.BA,T.BA       ;CHECK BUS ADDRESS CORRECT
2836 007634 001403      BEQ      3$              ;YES, CONTINUE
2837 007636 104023      ERROR   23              ;BUS ADDRESS INCORRECT
2838 007640 000137 010550      JMP      60$            ;CLEAR RK611
2839
2840 007644 023737 003502 003442 3$:      CMP      E.WC,T.WC       ;CHECK WORD COUNT CORRECT
2841 007652 001403      BEQ      4$              ;YES, CONTINUE
2842 007654 104024      ERROR   24              ;WORD COUNT INCORRECT
2843 007656 000137 010550      JMP      60$            ;CLEAR RK611
2844
2845 007662 012737 022040 003524 4$:      MOV      #ECCW!MEWD!DMD,E.MR1 ;LOAD EXPECTED MAINT REG. 1
2846 007670 005037 003622      CLR      BITCNT          ;INITIALIZE BIT COUNT
2847 007674 012700 000310      MOV      #200,R0         ;ISSUE 200 MAINT BITS
2848 007700 012762 000440 000026 5$:      MOV      #DMD!MCLK,RKMR1(R2)
2849 007706 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2850 007714 012762 000440 000026      MOV      #DMD!MCLK,RKMR1(R2)
2851 007722 012762 000040 000026      MOV      #DMD,RKMR1(R2)
2852 007730 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MAINT REG. 1
2853 007736 023737 003524 003464      CMP      E.MR1,T.MR1     ;CHECK MAINTENANCE REG. 1 CORRECT
2854 007744 001403      BEQ      6$              ;YES, CONTINUE
  
```

2855	007746	104025				ERROR	25		;MAINT REG 1 INCORRECT
2856	007750	000137	010550			JMP	60\$		;CLEAR RK611
2857									
2858	007754	005300			6\$:	DEC	R0		;CHECK IF READY FOR SECTOR PULSE
2859	007756	001350				BNE	5\$		;NO, GET NEXT BIT
2860	007760	016237	000000	003440		MOV	RKCS1(R2),T.CS1		;STORE COMMAND AND STATUS REG. 1
2861	007766	016237	000004	003444		MOV	RKBA(R2),T.BA		;STORE BUS ADDRESS REG
2862	007774	016237	000002	003442		MOV	RKWC(R2),T.WC		;STORE WORD COUNT REG
2863	010002	023737	003500	003440		CMP	E.CS1,T.CS1		;CHECK CS1 CORRECT
2864	010010	001403				BEQ	10\$		;YES, CONTINUE
2865	010012	104026				ERROR	26		;CS1 INCORRECT
2866	010014	000137	010550			JMP	60\$		;CLEAR RK611
2867									
2868	010020	023737	003504	003444	10\$:	CMP	E.BA,T.BA		;CHECK BUS ADDRESS CORRECT
2869	010026	001403				BEQ	11\$		;YES, CONTINUE
2870	010030	104027				ERROR	27		;BUS ADDRESS INCORRECT
2871	010032	000137	010550			JMP	60\$		;CLEAR RK611
2872									
2873	010036	023737	003502	003442	11\$:	CMP	E.WC,T.WC		;CHECK WORD COUNT REG. CORRECT
2874	010044	001403				BEQ	12\$		;YES, CONTINUE
2875	010046	104030				ERROR	30		;WORD COUNT INCORRECT
2876	010050	000137	010550			JMP	60\$		;CLEAR RK611
2877									
2878	010054	012700	000004		12\$:	MOV	#4,R0		;SIMULATE SECTOR PULSE
2879	010060	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)		
2880	010066	012762	000540	000026	13\$:	MOV	#DMD!MSP!MCLK,RKMR1(R2)		
2881	010074	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)		
2882	010102	005300				DEC	R0		
2883	010104	001370				BNE	13\$		
2884	010106	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
2885	010114	016237	000000	003440		MOV	RKCS1(R2),T.CS1		;STORE COMMAND AND STATUS REG. 1
2886	010122	016237	000004	003444		MOV	RKBA(R2),T.BA		;STORE BUS AND REG.
2887	010130	016237	000002	003442		MOV	RKWC(R2),T.WC		;STORE WORD COUNT REG.
2888	010136	023737	003500	003440		CMP	E.CS1,T.CS1		;CHECK COMMAND AND STATUS REG. 1 INCORRECT
2889	010144	001402				BEQ	15\$		;YES, CONTINUE
2890	010146	104026				ERROR	26		;CS1 INCORRECT
2891	010150	000577				BR	60\$		;CLEAR RK611
2892									
2893	010152	023737	003504	003444	15\$:	CMP	E.BA,T.BA		;CHECK BUS ADDRESS CORRECT
2894	010160	001402				BEQ	16\$		;YES, CONTINUE
2895	010162	104027				ERROR	27		;BUS ADDRESS INCORRECT
2896	010164	000571				BR	60\$		;CLEAR RK611
2897									
2898	010166	023737	003502	003442	16\$:	CMP	E.WC,T.WC		;CHECK WORD COUNT CORRECT
2899	010174	001402				BEQ	20\$		;YES, CONTINUE
2900	010176	104030				ERROR	30		;WORD COUNT INCORRECT
2901	010200	000563				BR	60\$		;CLEAR RK611
2902									
2903	010202	005077	003622		20\$:	CLR	BITCNT		;INITIALIZE BIT COUNT
2904	010206	012700	000310			MOV	#200,R0		;ISSUE 200 MAINT BITS
2905	010212	012762	000440	000026	21\$:	MOV	#DMD!MCLK,RKMR1(R2)		
2906	010220	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
2907	010226	012762	000440	000026		MOV	#DMD!MCLK,RKMR1(R2)		
2908	010234	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
2909	010242	016237	000026	003464		MOV	RKMR1(R2),T.MR1		;STORE MAINT REG. 1
2910	010250	023737	003524	003464		CMP	E.MR1,T.MR1		;CHECK MAINT REG. 1 CORRECT



2967  
2968  
2969  
2970  
2971  
2972  
2973  
2974  
2975  
2976  
2977  
2978  
2979  
2980  
2981  
2982  
2983  
2984  
2985  
2986  
2987  
2988  
2989  
2990  
2991  
2992  
2993  
2994  
2995  
2996  
2997  
2998  
2999  
3000  
3001  
3002  
3003  
3004  
3005  
3006  
3007  
3008  
3009  
3010  
3011  
3012  
3013  
3014  
3015  
3016  
3017  
3018  
3019  
3020  
3021  
3022

.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.  
*****
```

```
TST11: SCOPE  
MOV #100, $TIMES ;:DO 100. ITERATIONS  
MOV $BASE, R2 ;:LOAD RK611 BASE  
MOV #PATTERN, R3 ;:LOAD PATTERN ADDRESS  
MOV #8, R4 ;:LOAD PATTERN COUNT  
MOV #WRBUFF+2, E.BA ;:LOAD EXPECTED BUS ADDRESS  
MOV #WRHEAD, E.CS1 ;:LOAD EXPECTED CS1  
CLR E.WC ;:LOAD EXPECTED WORD COUNT  
MOV #1$, $LPERR ;:LOAD LOGP ON ERROR LOCATION FOR  
; SUBTEST LOOP  
  
1$:  
MOV #CCLR, RKCS1(R2) ;:CLEAR RK611  
MOV (R3), WRBUFF ;:LOAD BUFFER FOR WRITE HEADER  
MOV #DMD, RKMR1(R2) ;:PUT RK611 IN MAINT MODE  
MOV #777, RKDCYL(R2) ;:LOAD CYLINDER ADDRESS  
MOV #3400, RKDA(R2) ;:LOAD DISK ADDRESS  
MOV #WRBUFF, RKBA(R2) ;:LOAD BUS ADDRESS  
MOV #7, RKCS2(R2) ;:LOAD OTHER NUMBER  
MOV #-1, RKWC(R2) ;:LOAD WORD COUNT  
MOV #WRHEAD, RKCS1(R2) ;:ISSUE WRITE HEADER  
MOV #50.*4+2, R0 ;:ISSUE CLOCKS UNTIL READY FOR  
; INDEX PULSE  
  
2$:  
MOV #DMD!MCLK, RKMR1(R2)  
MOV #DMD, RKMR1(R2)  
DEC R0  
BNE 2$  
MOV #4, R0 ;:SIMULATE INDEX PULSE  
  
3$:  
MOV #DMD!MIND, RKMR1(R2)  
MOV #DMD!MIND!MCLK, RKMR1(R2)  
MOV #DMD!MIND, RKMR1(R2)  
DEC R0  
BNE 3$  
  
4$:  
MOV #DMD, RKMR1(R2)  
MOV #37, R0 ;:SIMULATE 1 NPR TRANSFER  
MOV #DMD!MCLK, RKMR1(R2)  
MOV #DMD, RKMR1(R2)  
DEC R0  
BNE 4$  
MOV RKCS1(R2), T.CS1 ;:STORE COMMAND AND STATUS REG. '  
MOV RKBA(R2), T.BA ;:STORE BUS ADDRESS REG
```

```

3023 011040 016237 000002 003442      MOV      RKWC(R2),T.WC      ;STORE WORD COUNT
3024 011046 012700 000024      MOV      #20.,R0          ;WAIT FOR OUTPUT READY
3025 011052 005300      5$:     DEC      R0
3026 011054 001376      BNE      5$
3027 011056 016237 000010 003450      MOV      RKCS2(R2),T.CS2   ;STORE COMMAND AND STATUS REG. 2
3028 011064 012737 000307 003510      MOV      #OR.IR.7,E.CS2   ;LOAD EXPECTED CS2
3029 011072 023737 003500 003440      CMP      E.CS1,T.CS1      ;CHECK CS1 CORRECT
3030 011100 001401      BEQ      6$               ;YES, CHECK CS2
3031 011102 104035      ERROR   35               ;CS1 INCORRECT
3032 011104 023737 003510 003450 6$:     CMP      E.CS2,T.CS2      ;CHECK CS2 CORRECT
3033 011112 001401      BEQ      7$               ;YES, CONTINUE
3034 011114 104036      ERROR   36               ;CS2 INCORRECT
3035 011116 023737 003504 003444 7$:     CMP      E.BA,T.BA        ;CHECK IF BUS ADDRESS INCREMENT OCCURRED
3036 011124 001401      BEQ      8$               ;YES, CONTINUE
3037 011126 104037      ERROR   37               ;BUS ADDRESS INCORRECT
3038 011130 023737 003502 003442 8$:     CMP      E.WC,T.WC        ;CHECK WORD COUNT REG CORRECT
3039 011136 001401      BEQ      9$               ;YES, CONTINUE
3040 011140 104040      ERROR   40               ;WORD COUNT INCORRECT
3041 011142 016237 000024 003462 9$:     MOV      RKDB(R2),T.DB     ;READ DATA BUFFER
3042 011150 011337 003522      MOV      (R3),E.DB        ;LOAD EXPECTED DATA BUFFER
3043 011154 023737 003522 003462      CMP      E.DB,T.DB        ;CHECK IF DATA CORRECT
3044 011162 001401      BEQ      15$              ;YES,CONTINUE
3045 011164 104041      ERROR   41               ;DATA BUFFER INCORRECT
3046 011166 016237 000000 003440 15$:    MOV      RKCS1(R2),T.CS1   ;STORE COMMAND AND STATUS REG. 1
3047 011174 016237 000010 003450      MOV      RKCS2(R2),T.CS2   ;STORE COMMAND AND STATUS REG. 2
3048 011202 012737 000107 003510      MOV      #IR!7,E.CS2      ;LOAD EXPECTED CS2
3049 011210 023737 003500 003440      CMP      E.CS1,T.CS1      ;CHECK COMMAND AND STATUS REG. 1 CORRECT
3050 011216 001401      BEQ      17$              ;YES, CONTINUE
3051 011220 104042      ERROR   42               ;CS1 INCORRECT
3052 011222 023737 003510 003450 7$:     CMP      E.CS2,T.CS2      ;CHECK COMMAND AND STATUS REG. 2 CORRECT
3053 011230 001401      BEQ      20$              ;YES, CONTINUE
3054 011232 104043      ERROR   43               ;CS2 INCORRECT
3055 011234 104415      20$:    SCOPE1                 ;CHECK IF LOOP ON ERROR
3056 011236 005723      TST      (R3)+             ;GENERATE ADDRESS OF NEXT CONFIG
3057 011240 005304      DEC      R4                ;CHECK IF ALL 8 CONFIGS TRIED
3058 011242 001000      BNE      TST12             ;:NO, TRY NEXT DATA PATTERN
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074 011244 000004      TST12:  SCOPE
3075 011246 012737 000144 001200      MOV      #100.,$TIMES     ;:DO 100. ITERATIONS
3076 011254 013702 001270      MOV      $BASE,R2         ;:LOAD RK611 BASE
3077 011260 012737 000001 001160      MOV      #1,$TMP0         ;:LOAD NUMBER OF WORDS FOR DATA TRANSFER
3078 011266 012704 000065      MOV      #65,R1           ;:LOAD ITERATION COUNT
  
```

```

*****
:TEST '2      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.
:
*****
  
```

```

3079 011272 012737 000027 003500      MOV      #WRHEAD,E.CS1      ;LOAD EXPECTED CS1
3080 011300 012737 011306 001110      MOV      #'$, $LPERR      ;LOAD LOOP ON ERROR LOCATION FOR
3081                                     ; SUBTEST LOOP
3082
3083 011306                                     1$:
3084 011306 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
3085 011314 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE
3086 011322 012762 064472 000004      MOV      #NPRBUF,RKBA(R2) ;LOAD BUS ADDRESS
3087 011330 012737 064472 003504      MOV      #NPRBUF,E.BA
3088 011336 013737 001160 003502      MOV      $TMP0,E.WC      ;LOAD WORD COUNT
3089 011344 005437 003502                NEG      E.WC
3090 011350 013762 003502 000002      MOV      E.WC,RKWC(R2)
3091 011356 012762 000027 000000      MOV      #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
3092 011364 012700 000312                MOV      #50.*4+2,R0      ;ISSUE CLOCKS UNTIL READY FOR
3093                                     ; INDEX PULSE
3094 011370 012762 000440 000026      2$: MOV      #DMD!MCLK,RKMR1(R2)
3095 011376 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3096 011404 005300                DEC      R0
3097 011406 001370                BNE     2$
3098 011410 012700 000004                MOV      #4,R0      ;SIMULATE INDEX PULSE
3099 011414 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
3100 011422 012762 000640 000026      3$: MOV      #DMD!MIND!MCLK,RKMR1(R2)
3101 011430 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
3102 011436 005300                DEC      R0
3103 011440 001370                BNE     3$
3104 011442 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3105 011450 012737 000300 003510      MOV      #IR!OR,E.CS2      ;LOAD EXPECTED CS2
3106 011456 012700 000045                4$: MOV      #37,R0      ;SIMULATE 1 NPR TRANSFER
3107 011462 012762 000440 000026      5$: MOV      #DMD!MCLK,RKMR1(R2)
3108 011470 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3109 011476 005300                DEC      R0
3110 011500 001370                BNE     5$
3111 011502 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3112 011510 016237 000004 003444      MOV      RKBA(R2),T.BA ;STORE BUS ADDRESS
3113 011516 016237 000002 003442      MOV      RKWC(R2),T.WC ;STORE WORD COUNT
3114 011524 022737 064472 003504      CMP      #NPRBUF,E.BA ;CHECK IF FIRST WORD
3115 011532 001004                BNE     7$ ;NO, STORE CS2
3116 011534 012700 000024                MOV      #20.,R0 ;WAIT FOR OUTPUT READY
3117 011540 005300                DEC      R0
3118 011542 001376                BNE     6$
3119 011544 016237 000010 003450      7$: MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3120 011552 062737 000002 003504      ADD      #2,E.BA ;INCREMENT WORD COUNT AND BUS ADD
3121 011560 005237 003502                INC      E.WC
3122 011564 023737 003500 003440      CMP      E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
3123 011572 001401                BEQ     8$ ;YES, CHECK CS2
3124 011574 104044                ERROR   44 ;CS1 INCORRECT
3125 011576 023737 003510 003450      8$: CMP      E.CS2,T.CS2 ;CHECK COMMAND STATUS REG. 2 CORRECT
3126 011604 001401                BEQ     9$ ;YES, CHECK BUSS ADDRESS REG.
3127 011606 104045                ERROR   45 ;CS2 INCORRECT
3128 011610 023737 003504 003444      9$: CMP      E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
3129 011616 001401                BEQ     10$ ;YES, CHECK WORD COUNT
3130 011620 104046                ERROR   46 ;BUS ADDRESS INCORRECT
3131 011622 023737 003502 003442      10$: CMP      E.WC,T.WC ;CHECK WORD COUNT CORRECT
3132 011630 001401                BEQ     11$ ;YES, CHECK IF ALL WORDS TRANSFERRED
3133 011632 104047                ERROR   47 ;WORD COUNT INCORRECT
3134 011634 005737 003502                11$: TST      E.WC ;CHECK IF FINISHED

```

3135	011640	001306				BNE	4\$		:NO, TRANSFER NEXT WORD
3136	011642	012700	000112			MOV	#2*37.,R0		:ISSUE ENOUC - CLOCKS FOR
3137	011646	012762	000440	000026	15\$	MOV	#DMD.MCLK,RKMR1(R2)		: 2 NPR TRANSFERS
3138	011654	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
3139	011662	005300				DEC	R0		
3140	011664	001370				BNE	15\$		
3141	011666	016237	000000	003440		MOV	RKCS1(R2),T.CS1		:STORE COMMAND AND STATUS REG. 1
3142	011674	016237	000010	003450		MOV	RKCS2(R2),T.CS2		:STORE COMMAND AND STATUS REG. 2
3143	011702	016237	000004	003444		MOV	RKBA(R2),T.BA		:STORE BUS ADDRESS REG.
3144	011710	016237	000002	003442		MOV	RKWC(R2),T.WC		:STORE WORD COUNT
3145	011716	023737	003500	003440		CMP	E.CS1,T.CS1		:CHECK COMMAND STATUS REG. 1 CORRECT
3146	011724	023737				BEQ	16\$		:YES, CHECK CS2
3147	011726	104050				ERROR	50		:CS1 INCORRECT
3148	011730	023737	003510	003450	16\$	CMP	E.CS2,T.CS2		:CHECK COMMAND STATUS REG. 2 CORRECT
3149	011736	001401				BEQ	17\$		:YES, CHECK BUS ADDRESS
3150	011740	104051				ERROR	51		:CS2 INCORRECT
3151	011742	023737	003504	003444	17\$	CMP	E.BA,T.BA		:CHECK BUS ADDRESS CORRECT
3152	011750	001401				BEQ	18\$		:YES, CHECK WORD COUNT
3153	011752	104052				ERROR	52		:BUS ADDRESS INCORRECT
3154	011754	023737	003502	003442	18\$	CMP	E.WC,T.WC		:CHECK WORD COUNT CORRECT
3155	011762	001401				BEQ	19\$		:YES, CONTINUE
3156	011764	104053				ERROR	53		:WORD COUNT INCORRECT
3157	011766	013701	001160		19\$	MOV	\$TMPO,R1		:LOAD NUMBER OF WORDS LOADED
3158	011772	012703	064472			MOV	#NPRBUFF,R3		:LOAD START ADDRESS
3159	011776	005037	003624			CLR	WRDCNT		:INITIALIZE WORD COUNT FOR PRINTOUT
3160	012002	016237	000024	003462	25\$	MOV	RKDB(R2),T.DB		:READ DATA BUFFER
3161	012010	012337	003522			MOV	(R3)+,E.DB		:LOAD EXPECTED DATA BUFFER
3162	012014	023737	003522	003462		CMP	E.DB,T.DB		:CHECK IF DATA CORRECT
3163	012022	001401				BEQ	26\$		:YES, CONTINUE
3164	012024	104054				ERROR	54		:DATA BUFFER INCORRECT
3165	012026	016237	000000	003440	26\$	MOV	RKCS1(R2),T.CS1		:STORE COMMAND AND STATUS REG. 1
3166	012034	016237	000010	003450		MOV	RKCS2(R2),T.CS2		:STORE COMMAND AND STATUS REG. 2
3167	012042	022701	000001			CMP	#1,R1		:CHECK IF LAST WORD IN SILO
3168	012046	001003				BNE	27\$		:NO, CONTINUE
3169	012050	012737	000100	003510		MOV	#1R,E.CS2		:LOAD EXPECTED CS2
3170	012056	023737	003500	003440	27\$	CMP	E.CS1,T.CS1		:CHECK COMMAND STATUS REG. 1 CORRECT
3171	012064	001401				BEQ	28\$		:YES, CONTINUE
3172	012066	104055				ERROR	55		:CS1 INCORRECT
3173	012070	023737	003510	003450	28\$	CMP	E.CS2,T.CS2		:CHECK COMMAND STATUS REG. 2 CORRECT
3174	012076	001401				BEQ	29\$		:YES, CONTINUE
3175	012100	104056				ERROR	56		:CS2 INCORRECT
3176	012102	005237	003624		29\$	INC	WRDCNT		:INCREMENT WORD COUNT
3177	012106	005301				DEC	R1		:DECREMENT WORDS READ
3178	012110	001334				BNE	25\$		:CHECK IF ALL WORDS READ
3179	012112	104415				SCOPI			:CHECK IF LOOP ON ERPOR
3180	012114	005237	001160			INC	\$TMPO		:INCREMENT WORDS READ FROM MEM
3181	012120	005304				DEC	R4		:CHECK IF FINISHED
3182	012122	001402				BEQ	TST13		:YES, GO ON TO NEXT TEST
3183	012124	000137	011306			JMP	1\$		:TRY NEXT WORD COUNT

3184  
 3185  
 3186 \*\*\*\*\*  
 3187 :TEST 13 SILO FILLING WITH NPR TRANSFERS  
 3188 :  
 3189 : CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER  
 3190 : IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06  
 : IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.



```

3191          : * SPECIFY A 66 WORD DATA TRANSFER, CLOCK IN ALL 66 WORDS
3192          : * INTO THE SILO. CHECK INPUT READY, OUTPUT READY, BUS
3193          : * ADDRESS, WORD COUNT, AND CONTENTS OF THE SILO.
3194          : *
3195          : *
3196 012130 000004 TST13: SCOPE
3197 012132 012737 000144 001200 MOV #100.,$TIMES ;;DO 100. ITERATIONS
3198 012140 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
3199 012144 012737 000027 003500 MOV #WRHEAD,E.CS1 ;LOAD EXPECTED CS1
3200 012152 012762 100000 000000 MOV #CLR,RKCS1(R2) ;CLEAR RK611
3201 012160 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE
3202 012166 012737 064474 003504 MOV #NPRBUF+2,E.BA ;LOAD BUS ADDRESS
3203 012174 012762 064472 000004 MOV #NPRBUF,RKBA(R2)
3204 012202 012737 177677 003502 MOV #-65.,E.WC ;LOAD WORD COUNT
3205 012210 012762 177676 000002 MOV #-66.,RKWC(R2)
3206 012216 012762 000027 000000 MOV #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
3207 012224 012700 000312 MOV #50.*4+2,R0 ;ISSUE CLOCKS UNTIL READY FOR
3208          : ; INDEX PULSE
3209 012230 012762 000440 000026 1$: MOV #DMD.MCLK,RKMR1(R2)
3210 012236 012762 000040 000026 MOV #DMD,RKMR1(R2)
3211 012244 005300 R0
3212 012246 001370 BNE 1$
3213 012250 012700 000004 MOV #4,R0 ;SIMULATE INDEX PULSE
3214 012254 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
3215 012262 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
3216 012270 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
3217 012276 005300 DEC R0
3218 012300 001370 BNE 2$
3219 012302 012762 000040 000026 MOV #DMD,RKMR1(R2)
3220 012310 012737 000300 003510 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3221 012316 012700 000045 000026 4$: MOV #37.,R0 ;SIMULATE 1 NPR TRANSFER
3222 012322 012762 000440 000026 5$: MOV #DMD.MCLK,RKMR1(R2)
3223 012330 012762 000040 000026 MOV #DMD,RKMR1(R2)
3224 012336 005300 DEC R0
3225 012340 001370 BNE 5$
3226 012342 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3227 012350 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
3228 012356 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WORD COUNT
3229 012364 022737 064474 003504 CMP #NPRBUF+2,E.BA ;CHECK IF FIRST WORD
3230 012372 001004 BNE 7$ ;NO, STORE CS2
3231 012374 012700 000024 MOV #20.,R0 ;WAIT FOR OUTPUT READY
3232 012400 005300 6$: DEC R0
3233 012402 001376 BNE 6$
3234 012404 016237 000010 003450 7$: MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3235 012412 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
3236 012420 001401 BEQ 8$ ;YES, CHECK CS2
3237 012422 104044 ERROR 44 ;CS1 INCORRECT
3238 012424 023737 003510 003450 8$: CMP E.CS2,T.CS2 ;CHECK COMMAND STATUS REG. 2 CORRECT
3239 012432 001401 BEQ 9$ ;YES, CHECK BUS ADDRESS
3240 012434 104045 ERROR 45 ;CS2 INCORRECT
3241 012436 023737 003504 003444 9$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
3242 012444 001401 BEQ 10$ ;YES, CHECK WORD COUNT
3243 012446 104046 ERROR 46 ;BUS ADDRESS INCORRECT
3244 012450 023737 003502 003442 10$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
3245 012456 001401 BEQ 11$ ;YES, CHECK IF ALL WORDS TRANSFERRED
3246 012460 104047 ERROR 47 ;WORD COUNT INCORRECT

```

3247	012462	062737	000002	003504	11\$:	ADD	#2,E.BA	:INCREMENT WORD COUNT AND BUS ADDRESS
3248	012470	005237	003502			INC	E.WC	:
3249	012474	100710				BMI	4\$	:CHECK IF FINISHED (NO, BRANCH)
3250	012476	001004				BNE	12\$	:CHECK IF LAST WORD
3251	012500	012737	000200	003510		MOV	#OR,E.CS2	:LOAD EXPECTED CS2
3252	012506	000703				BR	4\$	:PROCESS LAST WORD
3253								
3254	012510	005037	003502		12\$:	CLR	E.WC	:ADJUST EXPECTED WORD COUNT
3255	012514	162737	000002	003504		SUB	#2,E.BA	: AND BUS ADDRESS
3256	012522	012700	000112			MOV	#2*37.,RO	:ISSUE ENOUGH CLOCKS FOR
3257	012526	012762	000440	000026	15\$:	MOV	#DMD.MCLK,RKMR1(R2)	: 2 NPR TRANSFERS
3258	012534	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3259	012542	005300				DEC	RO	
3260	012544	001370				BNE	15\$	
3261	012546	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3262	012554	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3263	012562	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG
3264	012570	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT
3265	012576	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3266	012604	001401				BEQ	16\$	:YES, CHECK CS2
3267	012606	104050				ERROR	50	:CS1 INCORRECT
3268	012610	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3269	012616	001401				BEQ	17\$	:YES, CHECK BUS ADDRESS
3270	012620	104051				ERROR	51	:CS2 INCORRECT
3271	012622	023737	003504	003444	17\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3272	012630	001401				BEQ	18\$	:YES, CHECK WORD COUNT
3273	012632	104052				ERROR	52	:BUS ADDRESS INCORRECT
3274	012634	023737	003502	003442	18\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3275	012642	001401				BEQ	19\$	:YES, CHECK DATA
3276	012644	104053				ERROR	53	:WORD COUNT INCORRECT
3277	012646	012701	000102		19\$:	MOV	#66.,R1	:LOAD NUMBER OF WORDS LOADED
3278	012652	012703	064472			MOV	#NPRBUF,R3	:LOAD START ADDRESS
3279	012656	005037	003624			CLR	WRDCNT	:INITIALIZE WORD COUNT FOR PRINT OUT
3280	012662	012737	000300	003510		MOV	#IR!OR,E.CS2	:LOAD EXPECTED CS2
3281	012670	016237	000024	003462	25\$:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER
3282	012676	012337	003522			MOV	(R3)+,E.DB	:LOAD EXPECTED DATA BUFFER
3283	012702	023737	003522	003462		CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3284	012710	001401				BEQ	26\$	:YES, CONTINUE
3285	012712	104054				ERROR	54	:DATA BUFFER INCORRECT
3286	012714	016237	000000	003440	26\$:	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG 1
3287	012722	005737	003624			TST	WRDCNT	:CHECK IF FIRST WORD
3288	012726	001015				BNE	28\$	:NO, GET CS2
3289	012730	012700	000024			MOV	#20.,RO	:WAIT FOR INPUT READY TO SET
3290	012734	005300			27\$:	DEC	RO	
3291	012736	001376				BNE	27\$	
3292	012740	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3293	012746	022701	000001			CMP	#1,R1	:CHECK IF LAST WORD IN SILO
3294	012752	001003				BNE	28\$	:NO, CONTINUE
3295	012754	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
3296	012762	023737	003500	003440	28\$:	CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1 CORRECT
3297	012770	001401				BEQ	29\$	:YES, CHECK CS2
3298	012772	104055				ERROR	55	:CS1 INCORRECT
3299	012774	023737	003510	003450	29\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND AND STATUS REG. 2 CORRECT
3300	013002	001401				BEQ	30\$	:YES, GET NEXT WORD
3301	013004	104056				ERROR	56	:CS2 INCORRECT
3302	013006	005237	003624		30\$:	INC	WRDCNT	:INCREMENT WORD COUNT

```

3303 013012 005301      DEC      R1      ;DECREMENT WORDS READ
3304 013014 001325      BNE      25$     ;CHECK IF ALL WORDS READ
3305
3306
3307 :*****
3308 :*TEST 14      SILO CAPICITY WITH NPR TRANSFERS
3309 :*
3310 :*      CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
3311 :*      IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO AN RK06
3312 :*      IN 26 SECTOR FORMAT, CYLINDER 0.  HEAD 0, DRIVE 0.
3313 :*      SPECIFY A 68 WORD DATA TRANSFER.  CLOCK IN 66
3314 :*      WORDS INTO THE SILO.  MAKE SURE THAT SILO WILL STOP
3315 :*      FILLING AT 66 WORDS.  TAKE ONE WORD FROM SILO AND
3316 :*      CHECK IT.  CLOCK IN NEXT WORD.  MAKE SURE NO MORE
3317 :*      THAN ONE WORD IS CLOCKED IN THE SILO.  TAKE ONE
3318 :*      WORD FROM SILO AND CHECK IT.  CLOCK IN NEXT WORD.
3319 :*      CLOCK IN NEXT WORD.  MAKE SURE NO MORE THAN ONE WORD IS
3320 :*      CLOCKED IN THE SILO.  TAKE ONE WORD FROM SILO AND
3321 :*      CHECK IT.  ATTEMPT TO CLOCK IN NEXT WORD AND
3322 :*      MAKE SURE NO WORDS ARE CLOCK INTO SILO.  UNLOAD THE
3323 :*      SILO AND MAKE SURE ALL THE WORDS ARE CORRECT.
3324 :*****
3325 013016 000004      TEST14:  SCOPE
3326 013020 012737 000144 001200      MOV      #100.,$TIMES      ;:DO 100. ITERATIONS
3327 013026 013702 001270      MOV      $BASE,R2         ;:LOAD RK611 BASE
3328 013032 012737 000027 003500      MOV      #WRHEAD,E.CS1    ;:LOAD EXPECTED CS1
3329 013040 012762 100000 000000      MOV      #CCLR,RKCS1(R2)  ;:CLEAR RK611
3330 013046 012762 000040 000026      MOV      #DMD,RKMR1(R2)   ;:PUT RK611 IN MAINT MODE
3331 013054 012737 064474 003504      MOV      #NPRBUFF+2,E.BA  ;:LOAD BUS ADDRESS
3332 013062 012762 064472 000004      MOV      #NPRBUF,RKBA(R2)
3333 013070 012737 177675 003502      MOV      #-67.,E.WC       ;:LOAD WORD COUNT
3334 013076 012762 177674 000002      MOV      #-68.,RKWC(R2)
3335 013104 012762 000027 000C00      MOV      #WRHEAD,RKCS1(R2) ;:ISSUE WRITE HEADER
3336 013112 012700 000312      MOV      #50.*4+2,R0      ;:ISSUE CLOCKS UNTIL READY FOR
3337 :*      INDEX PULSE
3338 013116 012762 000440 000026 1$:  MOV      #DMD!MCLK,RKMR1(R2)
3339 013124 012762 000040 000026      MOV      #DMD,RKMR1(R2)
3340 013132 005300      DEC      R0
3341 013134 001370      BNE      1$
3342 013136 012700 000004      MOV      #4,R0           ;:SIMULATE INDEX PULSE
3343 013142 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
3344 013150 012762 000640 000026 2$:  MOV      #DMD!MIND!MCLK,RKMR1(R2)
3345 013156 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
3346 013164 005300      DEC      R0
3347 013166 001370      BNE      2$
3348 013170 012737 000040 000026      MOV      #DMD,RKMR1
3349 013176 012737 000300 003510      MOV      #JR!OR,E.CS2     ;:LOAD EXPECTED CS2
3350 013204 012700 000045      MOV      #37.,R0         ;:SIMULATE 1 NPR TRANSFER
3351 013210 012762 000440 000026 4$:  MOV      #DMD!MCLK,RKMR1(R2)
3352 013216 012762 000040 000026 5$:  MOV      #DMD,RKMR1(R2)
3353 013224 005300      DEC      R0
3354 013226 001370      BNE      5$
3355 013230 016237 000000 003440      MOV      RKCS1(R2),T.CS1  ;:STORE COMMAND AND REG. 1
3356 013236 016237 000004 003444      MOV      RKBA(R2),T.BA    ;:STORE BUS ADDRESS
3357 013244 016237 000002 003442      MOV      RKWC(R2),T.WC   ;:STORE WORD COUNT
3358 013252 022737 064474 003504      CMP      #NPRBUF+2,E.BA  ;:CHECK IF FIRST WORD

```

3359	013260	001004				BNE	7\$		:NO, STORE CS2
3360	013262	012700	000024			MOV	#20.,R0		:WAIT FOR OUTPUT READY
3361	013266	005300			6\$:	DEC	R0		
3362	013270	001376				BNE	6\$		
3363	013272	016237	000010	003450	7\$:	MOV	RKCS2(R2),T.CS2		:STORE COMMAND AND STATUS REG. 2
3364	013300	023737	003500	003440		CMP	E.CS1,T.CS1		:CHECK COMMAND STATUS REG. 1 CORRECT
3365	013306	001401				BEQ	8\$		:YES, CHECK CS2
3366	013310	104044				ERROR	44		:CS1 INCORRECT
3367	013312	023737	003510	003450	8\$:	CMP	E.CS2,T.CS2		:CHECK COMMAND STATUS REG. 2 CORRECT
3368	013320	001401				BEQ	9\$		:YES, CHECK BUS ADDRESS
3369	013322	104045				ERROR	45		:CS2 INCORRECT
3370	013324	023737	003504	003444	9\$:	CMP	E.BA,T.BA		:CHECK BUS ADDRESS CORRECT
3371	013332	001401				BEQ	10\$		:YES, CHECK WORD COUNT
3372	013334	104046				ERROR	46		:BUS ADDRESS INCORRECT
3373	013336	023737	003502	003442	10\$:	CMP	E.WC,T.WC		:CHECK WORD COUNT CORRECT
3374	013344	001401				BEQ	11\$		:YES, CHECK IF 1ST 65 WORDS TRANSFERRED
3375	013346	104047				ERROR	47		:WORD COUNT INCORRECT
3376	013350	062737	000002	003504	11\$:	ADD	#2,E.BA		:INCREMENT BUS ADD AND WORD COUNT
3377	013356	005237	003502			INC	E.WC		
3378	013362	022737	177776	003502		CMP	#-2,E.WC		:CHECK IF 65 WORDS IN SILO
3379	013370	101305				BHI	4\$		:NO, GET NEXT WORD
3380	013372	001004				BNE	12\$		:CHECK IF ALL 65 WORDS IN SILO
3381	013374	012737	000200	003510		MOV	#OR,E.CS2		:LOAD EXPECTED CS2
3382	013402	000700				BR	4\$		:PROCESS 66TH WORD
3383									
3384	013404	005337	003502		12\$:	DEC	E.WC		:ADJUST WORD COUNT AND
3385	013410	162737	000002	003504		SUB	#2,E.BA		: BUS ADDRESS
3386	013416	012701	000003			MOV	#3,R1		
3387	013422	005037	003624			CLR	WRDCNT		
3388	013426	012703	064472			MOV	#NPRBUF,R3		:LOAD START ADDRESS
3389	013432	012700	070112		13\$:	MOV	#2*37.,R0		:ISSUE ENOUGH CLOCKS FOR
3390	013436	012762	000440	000026	15\$:	MOV	#DMD!MCLK,RKMR1(R2)		: 2 NPR TRANSFERS
3391	013444	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
3392	013452	005300				DEC	R0		
3393	013454	001370				BNE	15\$		
3394	013456	016237	000000	003440		MOV	RKCS1(R2),T.CS1		:STORE COMMAND AND STATUS REG. 1
3395	013464	016237	000010	003450		MOV	RKCS2(R2),T.CS2		:STORE COMMAND AND STATUS REG. 2
3396	013472	016237	000004	003444		MOV	RKBA(R2),T.BA		:STORE BUS ADDRESS REG
3397	013500	016237	000002	003442		MOV	RKWC(R2),T.WC		:STORE WORD COUNT
3398	013506	023737	003500	003440		CMP	E.CS1,T.CS1		:CHECK COMMAND STATUS REG. 1
3399	013514	001401				BEQ	16\$		:YES, CHECK CS2
3400	013516	104050				ERROR	50		:CS1 INCORRECT
3401	013520	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2		:CHECK COMMAND STATUS REG. 2 CORRECT
3402	013526	001401				BEQ	17\$		:YES, CHECK BUS ADDRESS
3403	013530	104051				ERROR	51		:CS2 INCORRECT
3404	013532	023737	003504	003444	17\$:	CMP	E.BA,T.BA		:CHECK BUS ADD CORRECT
3405	013540	001401				BEQ	18\$		:YES, CHECK WORD COUNT
3406	013542	104052				ERROR	52		:BUS ADDRESS INCORRECT
3407	013544	023737	003502	003442	18\$:	CMP	E.WC,T.WC		:CHECK WORD COUNT CORRECT
3408	013552	001401				BEQ	19\$		:YES, READ 1 WORD FROM SILO
3409	013554	104053				ERROR	53		:WORD COUNT INCORRECT
3410	013556	012737	000300	003510	19\$:	MOV	#IR.OR,E.CS2		:LOAD EXPECT CS2
3411	013564	016237	000024	003462		MOV	RKDB(R2),T.DB		:STORE DATA BUFFER
3412	013572	012337	003522			MOV	(R3)+,E.DB		:LOAD EXPECTED DATA BUFFER
3413	013576	023737	003522	003462		CMP	E.DB,T.DB		:CHECK IF DATA CORRECT
3414	013604	001401				BEQ	25\$		:YES, CONTINUE

3415	013606	104054				ERROR	54		;DATA BUFFER INCORRECT
3416	013610	016237	000000	003440	25\$:	MOV	RKCS1(R2),T.CS1		;STORE COMMAND AND STATUS REG. 1
3417	013616	012700	000024			MOV	#20.,R0		;WAIT FOR OUTPUT READY
3418	013622	005300			26\$:	DEC	R0		
3419	013624	001376				BNE	26\$		
3420	013626	016237	000010	003450		MOV	RKCS2(R2),T.CS2		;STORE COMMAND AND STATUS REG. 2
3421	013634	023737	003500	003440		CMP	E.CS1,T.CS1		;CHECK IF COMMAND STATUS REG. 1 CORRECT
3422	013642	001401				BEQ	27\$		;YES, CHECK CS2
3423	013644	104055				ERROR	55		;CS1 INCORRECT
3424	013646	023737	003510	003450	27\$:	CMP	E.CS2,T.CS2		;CHECK IF COMMAND STATUS REG. 2 CORRECT
3425	013654	001401				BEQ	30\$		;YES, DO NPR TRANSFER
3426	013656	104056				ERROR	56		;CS2 INCORRECT
3427	013660	012700	000045		30\$:	MOV	#37.,R0		;CLOCK IN ONE WORD (NPR TRANSFER)
3428	013664	012762	000440	000026	31\$:	MOV	#DMD!MCLK,RKMR1(R2)		
3429	013672	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
3430	013700	005300				DEC	R0		
3431	013702	001370				BNE	31\$		
3432	013704	022701	000001			CMP	#1,R1		;CHECK IF 68TH WORD READ
3433	013710	001410				BEQ	32\$		;YES, NO NPR WILL TAKE PLACE
3434	013712	062737	000002	003504		ADD	#2,E.BA		;INCREMENT BUS ADD AND WORD COUNT
3435	013720	005237	003502			INC	E.WC		
3436	013724	012737	000200	003510		MOV	#OR,E.CS2		;LOAD EXPECTED CS2
3437	013732	016237	000000	003440	32\$:	MOV	RKCS1(R2),T.CS1		;STORE COMMAND AND STATUS REG. 1
3438	013740	016237	000010	003450		MOV	RKCS2(R2),T.CS2		;STORE COMMAND AND STATUS REG. 2
3439	013746	016237	000004	003444		MOV	RKBA(R2),T.BA		;STORE BUS ADDRESS
3440	013754	016237	000002	003442		MOV	RKWC(R2),T.WC		;STORE WORD COUNT
3441	013762	023737	003500	003440		CMP	E.CS1,T.CS1		;CHECK COMMAND STATUS REG. 1 CORRECT
3442	013770	001401				BEQ	33\$		;YES, CHECK CS2
3443	013772	104044				ERROR	44		;CS1 INCORRECT
3444	013774	023737	003510	003450	33\$:	CMP	E.CS2,T.CS2		;CHECK COMMAND STATUS REG. 2 CORRECT
3445	014002	001401				BEQ	34\$		;YES, CHECK BUS ADD
3446	014004	104045				ERROR	45		;CS2 INCORRECT
3447	014006	023737	003504	003444	34\$:	CMP	E.BA,T.BA		;CHECK BUS ADD CORRECT
3448	014014	001401				BEQ	35\$		;YES, CHECK WORD COUNT
3449	014016	104046				ERROR	46		;BUS ADD INCORRECT
3450	014020	023737	003502	003442	35\$:	CMP	E.WC,T.WC		;CHECK WORD COUNT CORRECT
3451	014026	001401				BEQ	36\$		;YES, CONTINUE
3452	014030	104047				ERROR	47		;WORD COUNT INCORRECT
3453	014032	005237	003624		36\$:	INC	WRDCNT		
3454	014036	005301				DEC	R1		;CHECK IF READ TO UNLOAD SILO
3455	014040	001402				BEQ	39\$		;YES, READ DATA; BUFFER
3456	014042	000137	013432			JMP	13\$		;NO, INPUT NEXT WORD
3457									
3458	014046	162737	000002	003504	39\$:	SUB	#2,E.BA		;ADJUST WORD COUNT AND BUS ADDRESS
3459	014054	005037	003502			CLR	E.WC		
3460	014060	012737	000300	003510		MOV	#1R,OR,E.CS2		;LOAD EXPECTED CS2
3461	014066	012701	000101			MOV	#65.,R1		;LOAD NUMBER OF WORDS LEFT
3462	014072	016237	000024	003462	40\$:	MOV	RKDB(R2),T.DB		;READ DATA BUFFER
3463	014100	012337	003522			MOV	(R3)+,E.DB		;LOAD EXPECTED DATA BUFFER
3464	014104	023737	003522	003462		CMP	E.DB,T.DB		;CHECK IF DATA CORRECT
3465	014112	001401				BEQ	41\$		;YES, CHECK CS1
3466	014114	104054				ERROR	54		;DATA BUFFER INCORRECT
3467	014116	016237	000000	003440	41\$:	MOV	RKCS1(R2),T.CS1		;STORE COMMAND AND STATUS REG. 1
3468	014124	022737	000002	003624		CMP	#2,WRDCNT		;CHECK IF FIRST WORDS
3469	014132	001004				BNE	43\$		;NO, DO NOT WAIT FOR INPUT READY
3470	014134	012700	000024			MOV	#20.,R0		;WAIT FOR INPUT READY

```

3471 014140 005300 42$: DEC R0
3472 014142 001376 BNE 42$
3473 014144 016237 000010 003450 43$: MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3474 014152 022701 000001 CMP #1,R1 ;CHECK IF LAST WORD
3475 014156 001003 BNE 44$ ;NO, CONTINUE
3476 014160 012737 000100 003510 MOV #IR,E.CS2 ;LOAD EXPECTED CS2
3477 014166 023737 003500 003440 44$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
3478 014174 001401 BEQ 45$ ;YES,CHECK CS2
3479 014176 104055 ERROR 55 ;CS1 INCORRECT
3480 014200 023737 003510 003450 45$: CMP E.CS2,T.CS2 ;CHECK COMMAND AND STATUS REG. 2 CORRECT
3481 014206 001401 BEQ 46$ ;YES, READ NEXT WORD ON SILO
3482 014210 104056 ERROR 56 ;CS2 INCORRECT
3483 014212 005237 003624 46$: INC WRDCNT ;INCREMENT WORD COUNT
3484 014216 005301 DEC R1 ;CHECK IF ALL WORDS READ
3485 014220 001324 BNE 40$ ;NO, READ NEXT WORD
3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499

```

```

*****
*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 WRCD.
*****

```

```

3499 014222 000004 TST15: SCOPE
3500 014224 012737 000144 001200 MOV #100.,$TIMES ;DO 100. ITERATIONS
3501 014232 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
3502 014236 012737 000027 003500 MOV #WRHEAD,E.CS1 ;LOAD EXPECTED CS1
3503 014244 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
3504 014252 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE
3505 014260 012737 064472 003504 MOV #NPRBUF,E.BA ;LOAD BUS ADDRESS
3506 014266 012762 064472 000004 MOV #NPRBUF,RKBA(R2)
3507 014274 012737 177677 003502 MOV #-65.,E.WC ;LOAD WORD COUNT
3508 014302 012762 177676 000002 MOV #-60.,RKWC(R2)
3509 014310 012762 000020 000010 MOV #BAI,RKCS2(R2) ;SET BUS ADDRESS INCREMENT INHIBIT
3510 014316 012762 000027 000000 MOV #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
3511 014324 012700 000312 MOV #50.*4+2,R0 ;ISSUE CLOCKS UNTIL READY FOR
3512 ; INDEX PULSE
3513 014330 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
3514 014336 012762 000040 000026 MOV #DMD,RKMR1(R2)
3515 014344 005300 DEC R0
3516 014346 001370 BNE 1$
3517 014350 012700 000004 MOV #4,R0 ;SIMULATE INDEX PULSE
3518 014354 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
3519 014362 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
3520 014370 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
3521 014376 005300 DEC R0
3522 014400 001370 BNE 2$
3523 014402 012762 000040 000026 MOV #DMD,RKMR1(R2)
3524 014410 012737 000320 003510 MOV #IR.OR!BAI,E.CS2 ;LOAD EXPECTED CS2
3525 014416 012700 000045 4$: MOV #37.,R0 ;SIMULATE 1 NPR TRANSFER
3526 014422 012762 000440 000026 5$: MOV #DMD!MCLK,RKMR1(R2)

```

3527	014430	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3528	014436	005300				DEC	R0	
3529	014440	001370				BNE	5\$	
3530	014442	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3531	014450	016237	000004	003444		MOV	RKBA(R2),1.BA	:STORE BUS ADDRESS
3532	014456	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT
3533	014464	022737	000101	003502		CMP	#65.,E.WC	:CHECK IF FIRST WORD
3534	014472	001004				BNE	7\$	:NO, STORE CS2
3535	014474	012700	000024			MOV	#20.,R0	:WAIT FOR OUTPUT READY
3536	014500	005300			6\$:	DEC	R0	
3537	014502	001376				BNE	6\$	
3538	014504	016237	000010	003450	7\$:	MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3539	014512	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3540	014520	001401				BEQ	8\$	:YES, CHECK CS2
3541	014522	104057				ERROR	57	:CS1 INCORRECT
3542	014524	023737	003510	003450	8\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3543	014532	001401				BEQ	9\$	:YES, CHECK BUS ADDRESS
3544	014534	104060				ERROR	60	:CS2 INCORRECT
3545	014536	023737	003504	003444	9\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS
3546	014544	001401				BEQ	10\$	:YES, CHECK WORD COUNT
3547	014546	104061				ERROR	61	:BUS ADDRESS INCORRECT
3548	014550	023737	003502	003442	10\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3549	014556	001401				BEQ	11\$	:YES, CHECK IF ALL WORDS TRANSFERRED
3550	014560	104062				ERROR	62	:WORD COUNT INCORRECT
3551	014562	005237	003502		11\$:	INC	E.WC	:INCREMENT WORD COUNT
3552	014566	100713				BMI	4\$	:CHECK IF FINISHED (NO, BRANCH)
3553	014570	001004				BNE	12\$	:CHECK IF LAST WORD
3554	014572	012737	000220	003510		MOV	#OR!BA1,E.CS2	:LOAD EXPECTED COMMAND STATUS REG. 2
3555	014600	000706				BR	4\$	:PROCESS THE LAST WORD
3556								
3557	014602	005037	003502		12\$:	CLR	E.WC	:ADJUST WORD COUNT
3558	014606	012700	000112			MOV	#2*37.,R0	:ISSUE ENOUGH CLOCKS FOR
3559	014612	012762	000440	000026	15\$:	MOV	#DMD!MCLK,RKMR1(R2)	: 2 NPR TRANSFERS
3560	014620	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3561	014626	005300				DEC	R0	
3562	014630	001370				BNE	15\$	
3563	014632	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3564	014640	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3565	014646	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG.
3566	014654	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT REG.
3567	014662	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3568	014670	001401				BEQ	16\$	:YES, CHECK CS2
3569	014672	104063				ERROR	63	:CS1 INCORRECT
3570	014674	023737	003510	003450	16\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3571	014702	001401				BEQ	17\$	:YES, CHECK BUS ADDRESS
3572	014704	104064				ERROR	64	:CS2 INCORRECT
3573	014706	023737	003504	003444	17\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3574	014714	001401				BEQ	18\$	:YES, CHECK WORD COUNT
3575	014716	104065				ERROR	65	:BUS ADDRESS INCORRECT
3576	014720	023737	003502	003442	18\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT CORRECT
3577	014726	001401				BEQ	19\$	:YES, CHECK DATA
3578	014730	104066				ERROR	66	:WORD COUNT INCORRECT
3579	014732	012701	000102		19\$:	MOV	#66.,R1	:LOAD NUMBERS OF WORDS LOADED
3580	014736	013737	064472	003522		MOV	NPRBUF,E.DB	:LOAD EXPECTED DATA BUFFER
3581	014744	005037	003624			CLR	WRD(NT	:INITIALIZE WORD COUNT FOR PRINT OUT
3582	014750	016237	000024	003462	25\$:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER



```

3583 014756 012737 000320 003510 MOV #IR.ORB!BAI,E.CS2 ;LOAD EXPECTED CS2
3584 014764 023737 003522 003462 CMP E.DB,T.DB ;CHECK IF DATA CORRECT
3585 014772 001401 BEQ 26$ ;YES, CONTINUE
3586 014774 104067 ERROR 67 ;DATA BUFFER INCORRECT
3587 014776 016237 000000 003440 26$: MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3588 015004 005737 003624 TST WRDCNT ;CHECK IF FIRST WORD
3589 015010 001015 BNE 28$ ;NO, GET CS2
3590 015012 012700 000024 MOV #20.,R0 ;WAIT FOR INPUT READY TO SET
3591 015016 005300 27$: DEC R0
3592 015020 001376 BNE 27$
3593 015022 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3594 015030 022701 000001 CMP #1,R1 ;CHECK IF LAST WORD
3595 015034 001003 BNE 28$ ;NO, CONTINUE
3596 015036 012737 000120 003510 MOV #IR.BAI,E.CS2 ;LOAD EXPECTED CS2
3597 015044 023737 003500 003440 28$: CMP E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG. 1 CORRECT
3598 015052 001401 BEQ 29$ ;YES, CHECK CS2
3599 015054 104070 ERROR 70 ;CS1 INCORRECT
3600 015056 023737 003510 003450 29$: CMP E.CS2,T.CS2 ;CHECK COMMAND STATUS REG 2 CORRECT
3601 015064 001401 BEQ 30$ ;YES, GET NEXT WORD
3602 015066 104071 ERROR 71 ;CS2 INCORRECT
3603 015070 005237 003624 30$: INC WRDCNT ;INCREMENT WORD COUNT
3604 015074 005301 DEC R1 ;DECREMENT WORDS READ
3605 015076 001324 BNE 25$ ;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
* (76000) AND MAKE SURE THE NON-EXISTENT MEMORY ERROR
* OCCURS IN THE RK611 CONTROLLER.
*****
  
```

```

3617 015100 000004 TEST16: SCOPE
3618 015102 012737 000144 001200 MOV #100.,$TIMES ;DO 100. ITERATIONS
3619 015110 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
3620 015114 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
3621 015122 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINT MODE
3622 015130 012737 160002 003504 MOV #160002,E.BA ;LOAD BUS ADDRESS
3623 015136 012762 160000 000004 MOV #160000,RKBA(R2)
3624 015144 012737 177677 003502 MOV #-65.,E.WC ;LOAD WORD COUNT
3625 015152 012762 177676 000002 MOV #-66.,RKWC(R2)
3626 015160 012737 101626 003500 MOV #CERR!RDY!BA16!BA17!WRHEAD<^C<GO>>,E.CS1
3627 015166 012762 001427 000000 MOV #BA17!BA16!WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
3628 015174 012700 000312 MOV #50.*4+2,R0 ;ISSUE CLOCKS UNTIL READY FOR
3629 3630 ; INDEX PULSE
3631 015200 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
3632 015206 012762 000040 000026 MOV #DMD,RKMR1(R2)
3633 015214 005300 DEC R0
3634 015216 001370 BNE 1$
3635 015220 012700 000004 MOV #4,R0 ;SIMULATE INDEX PULSE
3636 015224 012762 000240 000026 MOV #DMD.MIND,RKMR1(R2)
3637 015232 012762 000640 000026 2$: MOV #DMD.MIND!MCLK,RKMR1(R2)
3638 015240 012762 000240 000026 MOV #DMD.MIND,RKMR1(R2)
  
```



3639	015246	005300				DEC	R0	
3640	015250	001370				BNE	2\$	
3641	015252	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3642	015260	012700	000045			MOV	#37,R0	:SIMULATE 1 NPR TRANSFER
3643	015264	012762	000440	000026	5\$:	MOV	#DMD!MCLK,RKMR1(R2)	
3644	015272	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3645	015300	005300				DEC	R0	
3646	015302	001370				BNE	5\$	
3647	015304	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3648	015312	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3649	015320	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS
3650	015326	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT
3651	015334	016237	000014	003454		MOV	RKER(R2),T.ER	:STORE ERROR REG.
3652	015342	005037	003514			CLR	E.ER	:LOAD EXPECTED ERROR REG.
3653	015346	012737	004100	003510		MOV	#IR!NEM,E.CS2	:LOAD EXPECTED CS2
3654	015354	032737	000200	003450		BIT	#OR,T.CS2	
3655	015362	001403				BEQ	7\$	
3656	015364	052737	000200	003510		BIS	#OR,E.CS2	
3657	015372	023737	003500	003440	7\$:	CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3658	015400	001401				BEQ	8\$	:YES, CHECK CS2
3659	015402	104072				ERROR	72	:CS1 INCORRECT
3660	015404	023737	003510	003450	8\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND STATUS REG. 2 CORRECT
3661	015412	001401				BEQ	9\$	:YES, CHECK ERROR REG.
3662	015414	104073				ERROR	73	:CS2 INCORRECT
3663	015416	023737	003514	003454	9\$:	CMP	E.ER,T.ER	:CHECK ERROR REG CORRECT
3664	015424	001401				BEQ	10\$	:CHECK BUS ADDRESS
3665	015426	104074				ERROR	74	:ERROR REG INCORRECT
3666	015430	023737	003504	003444	10\$:	CMP	E.BA,T.BA	:CHECK BUS ADDRESS CORRECT
3667	015436	001401				BEQ	11\$	:YES, CHECK WORD COUNT
3668	015440	104075				ERROR	75	:BUS ADDRESS INCORRECT
3669	015442	023737	003502	003442	11\$:	CMP	E.WC,T.WC	:CHECK WORD COUNT REG.
3670	015450	001401				BEQ	12\$	:YES, CLEAR RK611
3671	015452	104076				ERROR	76	:WORD COUNT INCORRECT
3672	015454	012762	100000	000000	12\$:	MOV	#CLR,RKCS1(R2)	:CLEAR RK611
3673	015462	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3674	015470	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3675	015476	012737	000200	003500		MOV	#RDY,E.CS1	:LOAD EXPECTED CS1
3676	015504	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
3677	015512	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG. 1
3678	015520	001401				BEQ	15\$	:YES, CHECK CS2
3679	015522	104077				ERROR	77	:CS1 INCORRECT
3680	015524	023737	003510	003450	15\$:	CMP	E.CS2,T.CS2	:CHECK IF NEM CLEARED
3681	015532	001401				BEQ	TST17	:YES, GO ON TO NEXT TEST
3682	015534	104100				ERROR	100	:CS2 INCORRECT

```

3683
3684
3685 :*****
3686 :TEST 17      BUS ADDRESS BIT 16
3687 :
3688 :      CLEAR RK611 WITH A CONTROLLER CLEAR.  PUT CONTROLLER
3689 :      IN DIAGNOSTIC MODE.  ISSUE A WRITE HEADER TO  AN RK06
3690 :      IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
3691 :      SPECIFY A ONE WORD DATA TRANSFER FROM 200000.
3692 :      READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
3693 :      REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 177776.
3694 :      CHECK BUS ADDRESS AND WORD COUNT.
3695 :
3696 :*****

```

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

```
3695
3696
3697
3698
3699 015536 000004
3700 015540 012737 000144 001200
3701 015546 005737 043750
3702 015552 100004
3703 015554 022737 002000 044216
3704 015562 103417
3705 015564
3706 015564 012737 000001 001200
3707 015572 005227 177777
3708 015576 001007
3709 015600 104401 052443
3710 015604 013746 001220
3711 015610 104402
3712 015612 104401 052453
3713 015616 000137 016652
3714
3715 015622 013702 001270
3716 015626 012737 002000 172354
3717 015634 005237 177572
3718 015640 013737 064472 140000
3719 015646 005237 177572
3720 015652 012737 001110
3721
3722
3723 015660
3724 015660 012737 000000
3725 015666 012762 000026
3726 015674 012762 177777 000002
3727 015702 012737 000427 003500
3728 015710 012762 000427 000000
3729 015716 012700 000312
3730 015722 012762 000440 000026
3731 015730 012762 000040 000026
3732 015736 005300
3733 015740 001370
3734 015742 012700 000004
3735 015746 012762 000240 000026
3736 015754 012762 000640 000026
3737 015762 012762 000240 000026
3738 015770 005300
3739 015772 001370
3740 015774 012762 000040 000026
3741 016002 012700 000045
3742 016006 012762 000440 000026
3743 016014 012762 000040 000026
3744 016022 005300
3745 016024 001370
3746 016026 016237 000000 003440
3747 016034 016237 000004 003444
3748 016042 016237 000002 003442
3749 016050 012700 000024
3750 016054 005300
```

\*\*\*\*\*  
TST17: SCOPE  
MOV #100, \$TIMES ;DO 100. ITERATIONS  
TST \$KT11 ;CHECK FOR MEMORY MANAGEMENT  
BPL 1\$ ;NO, BYPASS TEST  
CMP #2000, \$LSTBK ;CHECK IF ENOUGH MEMORY  
BLO 2\$ ;YES, DO TEST  
1\$:  
MOV #1, \$TIMES ;FORCE INTERATION COUNT TO 1  
INC #-1 ;ONLY DO ONCE  
BNE 64\$ ;NO, GO TO NEXT TEST  
TYPE ,TSTBY1 ;TYPE TEST N BYPASSED  
MOV \$TESTN, -(SP) ;SAVE \$TESTN FOR TYPEOUT  
TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)  
TYPE ,TSTBY2  
64\$: JMP TST20 ;GO TO NEXT TEST  
2\$:  
MOV \$BASE, R2 ;LOAD K611 BASE  
MOV #2000, KIPAR6 ;LOAD PAGE ADDRESS 6 FOR DATA  
INC SRO ;TURN ON MEMORY MANAGEMENT  
MOV NPRBUF, 140000 ;LOAD WORD IN MEMORY  
CLR SRO ;TURN OFF MEMORY MANAGEMENT  
MOV #3\$, \$LPERR ;LOAD LOOP ON ERPOP LOCATION FOR  
; SUBTEST LOOP  
3\$:  
MOV #CLR, RKCS1(R2) ;CLEAR RK611  
MOV #DMD, RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE  
MOV #-1, RKWC(R2)  
MOV #BA16.WRHEAD, E.CS1 ;LOAD COMMAND  
MOV #BA16!WRHEAD, RKCS1(R2)  
MOV #50.\*4+2, R0 ;ISSUE ENOUGH CLOCKS UNTIL  
5\$: MOV #DMD!MCLK, RKMR1(R2) ; INDEX PULSE  
MOV #DMD, RKMR1(R2)  
DEC R0  
BNE 5\$  
MOV #4, R0 ;SIMULATE INDEX PULSE  
MOV #DMD.MIND, RKMR1(R2)  
6\$: MOV #DMD!MIND.MCLK, RKMR1(R2)  
MOV #DMD!MIND, RKMR1(R2)  
DEC R0  
BNE 6\$  
MOV #DMD, RKMR1(R2)  
7\$: MOV #37, R0 ;ISSUE 1 NPR TRANSFER  
MOV #DMD.MCLK, RKMR1(R2)  
MOV #DMD, RKMR1(R2)  
DEC R0  
BNE 7\$  
MOV RKCS1(R2), T.CS1 ;STORE COMMAND AND STATUS REG.  
MOV RKBA(R2), T.BA ;STORE BUS ADDRESS  
MOV RKWC(R2), T.WC ;STORE WOR  
8\$: MOV #20, R0 ;WAIT FOR OUTPUT READY  
DEC R0

3751	016056	001376				BNE	8\$	
3752	016060	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG. 2
3753	016066	012737	000300	003510		MOV	#1R.0R,E.CS2	:LOAD EXPECTED CS2
3754	016074	012737	000002	003504		MOV	#2,E.BA	:LOAD EXPECTED BUS ADDRESS
3755	016102	005037	003502			CLR	E.WC	:LOAD EXPECTED WORD COUNT
3756	016106	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3757	016114	001401				BEQ	10\$	:YES, CHECK CS2
3758	016116	104101				ERROR	101	:CS1 INCORRECT
3759	016120	023737	003510	003450	10\$:	CMP	E.CS2,T.CS2	:CHECK IF CS2 CORRECT
3760	016126	001401				BEQ	11\$	:YES, CHECK BUS ADDRESS CORRECT
3761	016130	104102				ERROR	102	:CS2 INCORRECT
3762	016132	023737	003504	003444	11\$:	CMP	E.BA,T.BA	:CHECK IF BUS ADD INCORRECT
3763	016140	001401				BEQ	12\$	:YES, CHECK WORD COUNT CORRECT
3764	016142	104103				ERROR	103	:BUS ADDRESS INCORRECT
3765	016144	023737	003502	003442	12\$:	CMP	E.WC,T.WC	:CHECK IF WORD COUNT CORRECT
3766	016152	001401				BEQ	13\$	:YES, CHECK DATA BUFFER
3767	016154	104104				ERROR	104	:WORD COUNT INCORRECT
3768	016156	016237	000024	003462	13\$:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER
3769	016164	013737	064472	003522		MOV	NPRBUF,E.DB	:LOAD EXPECTED DATA BUFFER
3770	016172	023737	003522	003462		CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3771	016200	001401				BFQ	15\$	:YES, CHECK IF LOOP ON ERROR
3772	016202	104105				ERROR	105	:DATA INCORRECT
3773	016204	104415			15\$:	SCOP1		:CHECK IF LOOP ON ERROR
3774	016206	012737	001777	172354		MOV	#2000-1,KIPAR6	:LOAD PAGE ADDRESS 6 FOR DATA
3775	016214	005237	177572			INC	SRO	:TURN ON MEMORY MANAGEMENT
3776	016220	013737	064472	140076		MOV	NPRBUF,140076	:LOAD WORDS IN MEMORY
3777	016226	013737	064474	140100		MOV	NPRBUF+2,140100	
3778	016234	005037	177572			CLR	SRO	:TURN OFF MEMORY MANAGEMENT
3779	016240	012737	016246	001110		MOV	#20\$,\$LPERR	:LOAD LOOP ON ERROR LOCATION FOR
3780								: SUBTEST LOOP
3781								
3782	016246				20\$:			
3783	016246	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611
3784	016254	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
3785	016262	012737	177777	003502		MOV	#-1,E.WC	:LOAD WORD COUNT REG.
3786	016270	012762	177776	000002		MOV	#-2,RKWC(R2)	
3787	016276	005037	003504			CLR	E.BA	:LOAD BUS ADDRESS
3788	016302	012762	177776	000004		MOV	#177776,RKBA(R2)	
3789	016310	012737	000427	003500		MOV	#BA16!WRHEAD,E.CS1	:LOAD COMMAND
3790	016316	012762	000027	000000		MOV	#WRHEAD,RKCS1(R2)	
3791	016324	012700	000312			MOV	#50.*4+2,R0	:ISSUE ENOUGH CLOCKS UNTIL
3792	016330	012762	000440	000026	21\$:	MOV	#DMD!MCLK,RKMR1(R2)	:INDEX PULSE
3793	016336	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3794	016344	005300				DEC	R0	
3795	016346	001370				BNE	21\$	
3796	016350	012700	000004			MOV	#4,R0	:SIMULATE INDEX PULSE
3797	016354	012762	000240	000026		MOV	#DMD.MIND,RKMR1(R2)	
3798	016362	012762	000640	000026	22\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
3799	016370	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
3800	016376	005300				DEC	R0	
3801	016400	001370				BNE	22\$	
3802	016402	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3803	016410	012701	000002			MOV	#2,R1	:ISSUE 2 NPR TRANSFERS
3804	016414	012700	000045		23\$:	MOV	#37,R0	
3805	016420	012762	000440	000026	24\$:	MOV	#DMD!MCLK,RKMR1(R2)	
3806	016426	012762	000040	000026		MOV	#DMD,RKMR1(R2)	

```

3807 016434 005300      DEC      R0
3808 016436 001370      BNE      24$
3809 016440 016237 000000 003440    MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3810 016446 016237 000004 003444    MOV      RKBA(R2),T.BA  ;STORE BUS ADDRESS REG.
3811 016454 016237 000002 003442    MOV      RKWC(R2),T.WC  ;STORE WORD COUNT
3812 016462 005737 003502      TST      E.WC           ;CHECK IF FIRST WORD
3813 016466 001404      BEQ      26$           ;NO, GET CS2
3814 016470 012700 000024      MOV      #20.,R0       ;WAIT FOR OUTPUT READ
3815 016474 005300      25$:    DEC      R0
3816 016476 001376      BNE      25$
3817 016500 016237 000010 003450 26$:    MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
3818 016506 012737 000300 003510    MOV      #IR.0R,E.CS2  ;LOAD EXPECTED CS2
3819 016514 023737 003500 003440    CMP      E.CS1,T.CS1   ;CHECK COMMAND STATUS REG 1 CORRECT
3820 016522 001401      BEQ      28$           ;YES, CHECK CS2
3821 016524 104101      ERROR    101          ;CS1 INCORRECT
3822 016526 023737 003510 003450 28$:    CMP      E.CS2,T.CS2   ;CHECK COMMAND STATUS REG 2 CORRECT
3823 016534 001401      BEQ      29$           ;YES, CHECK BUS ADDRESS
3824 016536 104102      ERROR    102          ;CS2 INCORRECT
3825 016540 023737 003504 003444 29$:    CMP      E.BA,T.BA     ;CHECK BUS ADDRESS CORRECT
3826 016546 001401      BEQ      30$           ;YES, CHECK WORD COUNT
3827 016550 104103      ERROR    103          ;BUS ADDRESS INCORRECT
3828 016552 023737 003502 003442 30$:    CMP      E.WC,T.WC     ;CHECK WORD COUNT CORRECT
3829 016560 001401      BEQ      31$           ;YES, GET TEXT WORD
3830 016562 104104      ERROR    104          ;WORD COUNT INCORRECT
3831 016564 062737 000002 003504 31$:    ADD      #2,E.BA       ;INCREMENT BUS ADDRESS AND
3832 016572 005237 003502      INC      E.WC          ;WORD COUNT
3833 016576 005301      DEC      R1            ;CHECK IF FINISHED
3834 016600 001305      BNE      23$           ;NO, GET SECOND WORD
3835 016602 012700 000002      MOV      #2,R0         ;LOAD COUNT AND ADDRESS WORD
3836 016606 012703 064472      MOV      #NPRBUF,R3    ;DATA COMPARE
3837 016612 016237 000024 003462 35$:    MOV      RKDB(R2),T.DB  ;READ DATA BUFFER
3838 016620 012337 003522      MOV      (R3)+,E.DB    ;GET EXPECTED DATA
3839 016624 023737 003522 003462    CMP      E.DB,T.DB     ;CHECK IF CORRECT
3840 016632 001401      BEQ      36$           ;YES, CHECK IF FINISHED
3841 016634 104105      ERROR    105          ;DATA BUFFER INCORRECT
3842 016636 005300      36$:    DEC      R0            ;CHECK IF FINISHED
3843 016640 001364      BNE      35$           ;NO READ SECOND WORD
3844 016642 104415      SCOP1     ;CHECK IF LOOP ON ERROR
3845 016644 012762 100000 000000    MOV      #CLR,RKCS1(R2) ;CLEAR RK611

```

```

3846
3847
3848 :*****
3849 :*TEST 20      BUS ADDRESS BIT 17
3850 :*
3851 :* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
3852 :* IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
3853 :* IN 26 SECTOR FORMAT, CYLINDER 0, HEADER 0, DRIVE 0.
3854 :* SPECIFY A ONE WORD DATA TRANSFER FROM 400000.
3855 :* READ THE SILO AND MAKE SURE RIGHT CONTENTS ARE READ.
3856 :* REPEAT FOR A TWO WORD TRANSFER FROM ADDRESS 377776.
3857 :* CHECK BUS ADDRESS AND WORD COUNT.
3858 :*
3859 :* NOTE: THIS TEST IS ONLY EXECUTED IF MORE THAN 64K
3860 :* OF MEMORY IS ON THE SYSTEM.
3861 :*****
3862 016652 000004 TST20: SCOPE

```

```

3863 016654 012737 000144 001200      MOV    #100.,$TIMES    ;;DO 100. ITERATIONS
3864 016662 005737 043750      TST    $KT11          ;CHECK FOR MEMORY MANAGEMENT
3865 016666 100004                BPL    1$             ;NO, BYPASS TEST
3866 016670 022737 004000 044216      CMP    #4000,$LSTBK   ;CHECK IF ENOUGH MEMORY
3867 016676 103417                BLO    2$             ;YES, DO TEST
3868 016700                        1$:
3869 016700 012737 000001 001200      MOV    #1,$TIMES     ;FORCE INTERATION COUNT TO 1
3870 016706 005227 177777      INC    #-i           ;ONLY DO ONCE
3871 016712 001007                RNE    64$           ;NO, GO TO NEXT TEST
3872 016714 104401 052443      TYPE  ,TSTBY1       ;TYPE TEST N BYPASSED
3873 016720 013746 001220      MOV    $TESTN,-(SP)  ;;SAVE $TESTN FOR TYPEOUT
3874 016724 104402                TYPOC                ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3875 016726 104401 052453      TYPE  ,TSTBY2
3876 016732 000137 017766      64$: JMP    TST21        ;GO TO NEXT TEST
3877
3878 016736 013702 001270      2$: MOV    $BASE,R2    ;LOAD K611 BASE
3879 016742 012737 004000 172354      MOV    #4000,KIPAR6 ;LOAD PAGE ADDRESS 6 FOR DATA
3880 016750 005237 177572      INC    SRO           ;TURN ON MEMORY MANAGEMENT
3881 016754 013737 064472 140000      MOV    NPRBUF,140000 ;LOAD WORD IN MEMORY
3882 016762 005037 177572      CLR    SRO           ;TURN OFF MEMORY MANAGEMENT
3883 016766 012737 016774 001110      MOV    #3$,$LPERR   ;LOAD LOOP ON ERROR LOCATION FOR
3884                                ; SUBTEST LOOP
3885
3886 016774                        3$:
3887 016774 012762 100000 000000      MOV    #CCLR,RKCS1(R2) ;CLEAR RK611
3888 017002 012762 000040 000026      MOV    #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
3889 017010 012762 177777 000002      MOV    #-1,RKWC(R2)
3890 017016 012737 001027 003500      MOV    #BA17!WRHEAD,E.CS1 ;LOAD COMMAND
3891 017024 012762 001027 000000      MOV    #BA17!WRHEAD,RKCS1(R2)
3892 017032 012700 000312                MOV    #50.*4+2,R0    ;ISSUE ENOUGH CLOCKS UNTIL
3893 017036 012762 070440 000026      5$: MOV    #DMD!MCLK,RKMR1(R2) ; INDEX PULSE
3894 017044 012762 000040 000026      MOV    #DMD,RKMR1(R2)
3895 017052 005300                DEC    R0
3896 017054 001370                BNE    5$
3897 017056 012700 000004                MOV    #4,R0          ;SIMULATE INDEX PULSE
3898 017062 012762 000240 000026      MOV    #DMD!MIND,RKMR1(R2)
3899 017070 012762 000640 000026      6$: MOV    #DMD!MIND!MCLK,RKMR1(R2)
3900 017076 012762 000240 000026      MOV    #DMD!MIND,RKMR1(R2)
3901 017104 005300                DEC    R0
3902 017106 001370                BNE    6$
3903 017110 012762 000040 000026      MOV    #DMD,RKMR1(R2)
3904 017116 012700 000045                MOV    #37.,R0        ;ISSUE 1 NPR TRANSFER
3905 017122 012762 000440 000026      7$: MOV    #DMD!MCLK,RKMR1(R2)
3906 017130 012762 000040 000026      MOV    #DMD,RKMR1(R2)
3907 017136 005300                DEC    R0
3908 017140 001370                BNE    7$
3909 017142 016237 000000 003440      MOV    RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
3910 017150 016237 000004 003444      MOV    RKBA(R2),T.BA  ;STORE BUS ADDRESS
3911 017156 016237 000002 003442      MOV    RKWC(R2),T.WC  ;STORE WOR
3912 017164 012700 000024                MOV    #20.,R0        ;WAIT FOR OUTPUT READY
3913 017170 005300                        8$: DEC    R0
3914 017172 001376                BNE    8$
3915 017174 016237 000010 003450      MOV    RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
3916 017202 012737 000300 003510      MOV    #IR.OR,E.CS2  ;LOAD EXPECTED CS2
3917 017210 012737 000002 003504      MOV    #2,E.BA       ;LOAD EXPECTED BUS ADDRESS
3918 017216 005037 003502      CLR    E.WC          ;LOAD EXPECTED WORD COUNT

```

3919	017222	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND STATUS REG. 1 CORRECT
3920	017230	001401				BEQ	10\$	:YES, CHECK CS2
3921	017232	104101				ERROR	101	:CS1 INCORRECT
3922	017234	023737	003510	003450	10\$:	CMP	E.CS2,T.CS2	:CHECK IF CS2 CORRECT
3923	017242	001401				BEQ	11\$	:YES, CHECK BUS ADDRESS CORRECT
3924	017244	104102				ERROR	102	:CS2 INCORRECT
3925	017246	023737	003504	003444	11\$:	CMP	E.BA,T.BA	:CHECK IF BUS ADD INCORRECT
3926	017254	001401				BEQ	12\$	:YES, CHECK WORD COUNT CORRECT
3927	017256	104103				ERROR	103	:BUS ADDRESS INCORRECT
3928	017260	023737	003502	003442	12\$:	CMP	E.WC,T.WC	:CHECK IF WORD COUNT CORRECT
3929	017266	001401				BEQ	13\$	:YES, CHECK DATA BUFFER
3930	017270	104104				ERROR	104	:WORD COUNT INCORRECT
3931	017272	016237	000024	003462	13\$:	MOV	RKDB(R2),T.DB	:READ DATA BUFFER
3932	017300	013737	064472	003522		MOV	NPRBUF,E.DB	:LOAD EXPECTED DATA BUFFER
3933	017306	023737	003522	003462		CMP	E.DB,T.DB	:CHECK IF DATA CORRECT
3934	017314	001401				BEQ	15\$	:YES, CHECK IF LOOP ON ERROR
3935	017316	104105				ERROR	105	:DATA INCORRECT
3936	017320	104415			15\$:	SCOP1		:CHECK IF LOOP ON ERROR
3937	017322	012737	003777	172354		MOV	#4000-1,KIPAR6	:LOAD PAGE ADDRESS 6 FOR DATA
3938	017330	005237	177572			INC	SRO	:TURN ON MEMORY MANAGEMENT
3939	017334	013737	064472	140076		MOV	NPRBUF,140076	:LOAD WORDS IN MEMORY
3940	017342	013737	064474	140100		MOV	NPRBUF+2,140100	
3941	017350	005037	177572			CLR	SRO	:TURN OFF MEMORY MANAGEMENT
3942	017354	012737	017362	001110		MOV	#20\$,\$LPERR	:LOAD LOOP ON ERROR LOCATION FOR
3943								: SUBTEST LOOP
3944								
3945	017362				20\$:			
3946	017362	012762	100000	000000		MOV	#CCLR,RKCS1(R2)	:CLEAR RK611
3947	017370	012762	000040	000026		MOV	#DMD,RKMR1(R2)	:PUT RK611 IN DIAGNOSTIC MODE
3948	017376	012737	177777	003502		MOV	#-1,E.WC	:LOAD WORD COUNT REG.
3949	017404	012762	177776	000002		MOV	#-2,RKWC(R2)	
3950	017412	005037	003504			CLR	E.BA	:LOAD BUS ADDRESS
3951	017416	012762	177776	000004		MOV	#177776,RKBA(R2)	
3952	017424	012737	001027	003500		MOV	#BA17!WRHEAD,E.CS1	:LOAD COMMAND
3953	017432	012762	000427	000000		MOV	#BA16!WRHEAD,RKCS1(R2)	
3954	017440	012700	000312			MOV	#50.*4+2,R0	:ISSUE ENOUGH CLOCKS UNTIL
3955	017444	012762	000440	000026	21\$:	MOV	#DMD!MCLK,RKMR1(R2)	: INDEX PULSE
3956	017452	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3957	017460	005300				DEC	R0	
3958	017462	001370				BNE	21\$	
3959	017464	012700	000004			MOV	#4,R0	:SIMULATE INDEX PULSE
3960	017470	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
3961	017476	012762	000640	000026	22\$:	MOV	#DMD!MIND.MCLK,RKMR1(R2)	
3962	017504	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
3963	017512	005300				DEC	R0	
3964	017514	001370				BNE	22\$	
3965	017516	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3966	017524	012701	000002			MOV	#2,R1	:ISSUE 2 NPR TRANSFERS
3967	017530	012700	000045		23\$:	MOV	#37,R0	
3968	017534	012762	000440	000026	24\$:	MOV	#DMD!MCLK,RKMR1(R2)	
3969	017542	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
3970	017550	005300				DEC	R0	
3971	017552	001370				BNE	24\$	
3972	017554	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG. 1
3973	017562	016237	000004	003444		MOV	RKBA(R2),T.BA	:STORE BUS ADDRESS REG.
3974	017570	016237	000002	003442		MOV	RKWC(R2),T.WC	:STORE WORD COUNT

```

3975 017576 005737 003502      TST      E.WC      ;CHECK IF FIRST WORD
3976 017602 001404      BEQ      26$      ;NO, GET CS2
3977 017604 012700 000024      MOV      #20.,R0  ;WAIT FOR OUTPUT READY
3978 017610 005300      25$:    DEC      R0
3979 017612 001376      BNE      25$
3980 017614 016237 000010 003450 26$:    MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
3981 017622 012737 000300 003510  MOV      #IR!OR,E.CS2 ;LOAD EXPECTED CS2
3982 017630 023737 003500 003440  CMP      E.CS1,T.CS1 ;CHECK COMMAND STATUS REG 1 CORRECT
3983 017636 001401      BEQ      28$      ;YES, CHECK CS2
3984 017640 104101      ERROR   101      ;CS1 INCORRECT
3985 017642 023737 003510 003450 28$:    CMP      E.CS2,T.CS2 ;CHECK COMMAND STATUS REG 2 CORRECT
3986 017650 001401      BEQ      29$      ;YES, CHECK BUS ADDRESS
3987 017652 104102      ERROR   102      ;CS2 INCORRECT
3988 017654 023737 003504 003444 29$:    CMP      E.BA,T.BA  ;CHECK BUS ADDRESS CORRECT
3989 017662 001401      BEQ      30$      ;YES, CHECK WORD COUNT
3990 017664 104103      ERROR   103      ;BUS ADDRESS INCORRECT
3991 017666 023737 003502 003442 30$:    CMP      E.WC,T.WC  ;CHECK WORD COUNT CORRECT
3992 017674 001401      BEQ      31$      ;YES, GET TEXT WORD
3993 017676 104104      ERROR   104      ;WORD COUNT INCORRECT
3994 017700 062737 000002 003504 31$:    ADD      #2,E.BA  ;INCREMENT BUS ADDRESS AND
3995 017706 005237 003502      INC      E.WC      ;WORD COUNT
3996 017712 005301      DEC      R1        ;CHECK IF FINISHED
3997 017714 001305      BNE      23$      ;NO, GET SECOND WORD
3998 017716 012700 000002      MOV      #2,R0    ;LOAD COUNT AND ADDRESS WORD
3999 017722 012703 064472      MOV      #NPRBUF,R3 ;DATA COMPARE
4000 017726 016237 000024 003462 35$:    MOV      RKDB(R2),T.DB ;READ DATA BUFFER
4001 017734 012337 003522      MOV      (R3)+,E.DB ;GET EXPECTED DATA
4002 017740 023737 003522 003462  CMP      E.DB,T.DB  ;CHECK IF CORRECT
4003 017746 001401      BEQ      36$      ;YES, CHECK IF FINISHED
4004 017750 104105      ERROR   105      ;DATA BUFFER INCORRECT
4005 017752 005300      36$:    DEC      R0        ;CHECK IF FINISHED
4006 017754 001364      BNE      35$      ;NO READ SECOND WORD
4007 017756 104415      SCOP1   ;CHECK IF LOOP ON ERROR
4008 017760 012762 100000 000000  MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024

```

```

*****
: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, HEADER 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.

```

```

*****

```

```

4025 017766 000004      TST21: SCOPE
4026 017770 012737 000144 001200  MOV      #100.,$TIMES ;DO 100. ITERATIONS
4027 017776 005737 043750      TST      $KT11     ;CHECK FOR MEMORY MANAGEMENT
4028 020002 100004      BPL      1$        ;NO, BYPASS TEST
4029 020004 022737 006000 044216  CMP      #6000,$LSTBK ;CHECK IF ENOUGH MEMORY
4030 020012 103417      BLO      2$        ;YES, DO TEST

```



```

4031 020014
4032 020014 012737 000001 001200 1$: MOV #1,$TIMES ;FORCE INTERATION COUNT TO 1
4033 020022 005227 177777 INC #-1 ;ONLY DO ONCE
4034 020026 001007 BNE 64$ ;NO, GO TO NEXT TEST
4035 020030 104401 052443 TYPE ,TSTBY1 ;TYPE TEST N BYPASSED
4036 020034 013746 001220 MOV $TESTN,-(SP) ;;SAVE $TESTN FOR TYPEOUT
4037 020040 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
4038 020042 104401 052453 TYPE ,TSTBY2
4039 020046 000137 021102 64$: JMP $TST2 ;GO TO NEXT TEST
4040
4041 020052 013702 001270 2$: MOV $BASE,R2 ;LOAD K611 BASE
4042 020056 012737 006000 172354 MOV #6000,KIPAR6 ;LOAD PAGE ADDRESS 6 FOR DATA
4043 020064 005237 177572 INC SR0 ;TURN ON MEMORY MANAGEMENT
4044 020070 013737 064472 140000 MOV NPRBUF,140000 ;LOAD WORD IN MEMORY
4045 020076 005037 177572 CLR SR0 ;TURN OFF MEMORY MANAGEMENT
4046 020102 012737 020110 001110 MOV #3$,$LPERR ;LOAD LOOP ON ERROR LOCATION FOR
4047 ; SUBTEST LOOP
4048
4049 020110 3$:
4050 020110 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4051 020116 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4052 020124 012762 177777 000002 MOV #-1,RKWC(R2)
4053 020132 012737 001427 003500 MOV #BA16!BA17!WRHEAD,E.CS1 ;LOAD COMMAND
4054 020140 012762 001427 000000 MOV #BA16!BA17!WRHEAD,RKCS1(R2)
4055 020146 012700 000312 MOV #50,*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL
4056 020152 012762 000440 000026 5$: MOV #DMD!MCLK,RKMR1(R2) ; INDEX PULSE
4057 020160 012762 000040 000026 MOV #DMD,RKMR1(R2)
4058 020166 005300 DEC R0
4059 020170 001370 BNE 5$
4060 020172 012700 000004 MOV #4,R0 ;SIMULATE INDEX PULSE
4061 020176 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
4062 020204 012762 000640 000026 6$: MOV #DMD!MIND!MCLK,RKMR1(R2)
4063 020212 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
4064 020220 005300 DEC R0
4065 020222 001370 BNE 6$
4066 020224 012762 000040 000026 MOV #DMD,RKMR1(R2)
4067 020232 012700 000045 MOV #37.,R0 ;ISSUE 1 NPR TRANSFER
4068 020236 012762 000440 000026 7$: MOV #DMD.MCLK,RKMR1(R2)
4069 020244 012762 000040 000026 MOV #DMD,RKMR1(R2)
4070 020252 005300 DEC R0
4071 020254 001370 BNE 7$
4072 020256 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4073 020264 016237 000004 003444 MOV RKBA(R2),T.BA ;STORE BUS ADDRESS
4074 020272 016237 000002 003442 MOV RKWC(R2),T.WC ;STORE WOR
4075 020300 012700 000024 MOV #20.,R0 ;WAIT FOR OUTPUT READY
4076 020304 005300 8$: DEC R0
4077 020306 001376 BNE 8$
4078 020310 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4079 020316 012737 000300 003510 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4080 020324 012737 000002 003504 MOV #2,E.BA ;LOAD EXPECTED BUS ADDRESS
4081 020332 005037 003502 CLR E.WC ;LOAD EXPECTED WORD COUNT
4082 020336 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND STATUS REG. 1 CORRECT
4083 020344 001401 BEQ 10$ ;YES, CHECK CS2
4084 020346 104101 ERROR 101 ;CS1 INCORRECT
4085 020350 023737 003510 003450 10$: CMP E.CS2,T.CS2 ;CHECK IF CS2 CORRECT
4086 020356 001401 BEQ 11$ ;YES, CHECK BUS ADDRESS CORRECT
  
```



4087	020360	104102				ERROR	102		:CS2 INCORRECT
4088	020362	023737	003504	003444	11\$:	CMP	E.BA,*.BA		:CHECK IF BUS ADD INCORRECT
4089	020370	001401				BEQ	12\$		:YES, CHECK WORD COUNT CORRECT
4090	020372	104103				ERROR	103		:BUS ADDRESS INCORRECT
4091	020374	023737	003502	003442	12\$:	CMP	E.WC,T.WC		:CHECK IF WORD COUNT CORRECT
4092	020402	001401				BEQ	13\$		:YES, CHECK DATA BUFFER
4093	020404	104104				ERROR	104		:WORD COUNT INCORRECT
4094	020406	016237	000024	003462	13\$:	MOV	RKDB(R2),*.DB		:READ DATA BUFFER
4095	020414	013737	064472	003522		MOV	NPRBUF,E.DB		:LOAD EXPECTED DATA BUFFER
4096	020422	023737	003522	003462		CMP	E.DB,T.DB		:CHECK IF DATA CORRECT
4097	020430	001401				BEQ	15\$		:YES, CHECK IF LOOP ON ERROR
4098	020432	104105				ERROR	105		:DATA INCORRECT
4099	020434	104415			15\$:	SCOP1			:CHECK IF LOOP ON ERROR
4100	020436	012737	005777	172354		MOV	#6000-1,KIPAR6		:LOAD PAGE ADDRESS 6 FOR DATA
4101	020444	005237	177572			INC	SRO		:TURN ON MEMORY MANAGEMENT
4102	020450	013737	064472	140076		MOV	NPRBUF,140076		:LOAD WORDS IN MEMORY
4103	020456	013737	064474	140100		MOV	NPRBUF+2,140100		
4104	020464	005037	177572			CLR	SRO		:TURN OFF MEMORY MANAGEMENT
4105	020470	012737	020476	001110		MOV	#20\$,\$LPERR		:LOAD LOOP ON ERROR LOCATION FOR
4106									: SUBTEST LOOP
4107									
4108	020476				20\$:				
4109	020476	012762	100000	000000		MOV	#CCLR,RKCS1(R2)		:CLEAR RK611
4110	020504	012762	000040	000026		MOV	#DMD,RKMR1(R2)		:PUT RK611 IN DIAGNOSTIC MODE
4111	020512	012737	177777	003502		MOV	#-1,E.WC		:LOAD WORD COUNT REG.
4112	020520	012762	177776	000002		MOV	#-2,RKWC(R2)		
4113	020526	005037	003504			CLR	E.BA		:LOAD BUS ADDRESS
4114	020532	012762	177776	000004		MOV	#177776,RKBA(R2)		
4115	020540	012737	001427	003500		MOV	#BA16!BA17!WRHEAD,E.CS1		:LOAD COMMAND
4116	020546	012762	001027	000000		MOV	#BA17!WRHEAD,RKCS1(R2)		
4117	020554	012700	000312			MOV	#50.*4+2,R0		:ISSUE ENOUGH CLOCKS UNTIL
4118	020560	012762	000440	000026	21\$:	MOV	#DMD!MCLK,RKMR1(R2)		: INDEX PULSE
4119	020566	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
4120	020574	005300				DEC	R0		
4121	020576	001370				BNE	21\$		
4122	020600	012700	000004			MOV	#4,R0		:SIMULATE INDEX PULSE
4123	020604	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)		
4124	020612	012762	000640	000026	22\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)		
4125	020620	012762	000240	000026		MOV	#DMD.MIND,RKMR1(R2)		
4126	020626	005300				DEC	R0		
4127	020630	001370				BNE	22\$		
4128	020632	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
4129	020640	012701	000002			MOV	#2,R1		:ISSUE 2 NPR TRANSFERS
4130	020644	012700	000045		23\$:	MOV	#37,R0		
4131	020650	012762	000440	000026	24\$:	MOV	#DMD!MCLK,RKMR1(R2)		
4132	020656	012762	000040	000026		MOV	#DMD,RKMR1(R2)		
4133	020664	005300				DEC	R0		
4134	020666	001370				BNE	24\$		
4135	020670	016237	000000	003440		MOV	RKCS1(R2),T.CS1		:STORE COMMAND AND STATUS REG. 1
4136	020676	016237	000004	003444		MOV	RKBA(R2),T.BA		:STORE BUS ADDRESS REG.
4137	020704	016237	000002	003442		MOV	RKWC(R2),T.WC		:STORE WORD COUNT
4138	020712	005737	003502			TST	E.WC		:CHECK IF FIRST WORD
4139	020716	001404				BEQ	26\$		:NO, GET CS2
4140	020720	012700	000024			MOV	#20.,R0		:WAIT FOR OUTPUT READY
4141	020724	005300			25\$:	DEC	R0		
4142	020726	001376				BNE	25\$		

```

4143 020730 016237 000010 003450 26$: MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
4144 020736 012737 000300 003510 MOV #IR!OR,E.CS2 ;LOAD EXPECTED CS2
4145 020744 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK COMMAND STATUS REG 1 CORRECT
4146 020752 001401 BEQ 28$ ;YES, CHECK CS2
4147 020754 104101 ERROR 101 ;CS1 INCORRECT
4148 020756 023737 003510 003450 28$: CMP E.CS2,T.CS2 ;CHECK COMMAND STATUS REG 2 CORRECT
4149 020764 001401 BEQ 29$ ;YES, CHECK BUS ADDRESS
4150 020766 104102 ERROR 102 ;CS2 INCORRECT
4151 020770 023737 003504 003444 29$: CMP E.BA,T.BA ;CHECK BUS ADDRESS CORRECT
4152 020776 001401 BEQ 30$ ;YES, CHECK WORD COUNT
4153 021000 104103 ERROR 103 ;BUS ADDRESS INCORRECT
4154 021002 023737 003502 003442 30$: CMP E.WC,T.WC ;CHECK WORD COUNT CORRECT
4155 021010 001401 BEQ 31$ ;YES, GET TEXT WORD
4156 021012 104104 ERROR 104 ;WORD COUNT INCORRECT
4157 021014 062737 000002 003504 31$: ADD #2,E.BA ;INCREMENT BUS ADDRESS AND
4158 021022 005237 003502 INC E.WC ;WORD COUNT
4159 021026 005301 DEC R1 ;CHECK IF FINISHED
4160 021030 001305 BNE 23$ ;NO, GET SECOND WORD
4161 021032 012700 000002 MOV #2,R0 ;LOAD COUNT AND ADDRESS WORD
4162 021036 012703 064472 MOV #NPRBUF,R3 ;DATA COMPARE
4163 021042 016237 000024 003462 35$: MOV RKDB(R2),T.DB ;READ DATA BUFFER
4164 021050 012337 003522 MOV (R3)+,E.DB ;GET EXPECTED DATA
4165 021054 023737 003522 003462 CMP E.DB,T.DB ;CHECK IF CORRECT
4166 021062 001401 BEQ 36$ ;YES, CHECK IF FINISHED
4167 021064 104105 ERROR 105 ;DATA BUFFER INCORRECT
4168 021066 005300 36$: DEC R0 ;CHECK IF FINISHED
4169 021070 001364 BNE 35$ ;NO READ SECOND WORD
4170 021072 104415 SCOP1 ;CHECK IF LOOP ON ERROR
4171 021074 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
  
```

4172  
4173  
4174  
4175  
4176  
4177  
4178  
4179  
4180  
4181  
4182  
4183  
4184

```

*****
*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).
*****
  
```

```

4185 021102 000004 TST22: SCOP1
4186 021104 012737 000144 001200 MOV #100, $TIMES ;DO 100. ITERATIONS
4187 021112 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
4188 021116 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4189 021124 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
4190 021132 012762 171676 000002 MOV #-66, RKWC(R2) ;LOAD WORD COUNT
4191 021140 012737 177677 003502 MOV #-65, E.WC
4192 021146 012762 064472 000004 MOV #NPRBUF,RKBA(R2) ;LOAD BUS ADDRESS
4193 021154 012737 064474 003504 MOV #NPRBUF+2,E.BA
4194 021162 012737 010027 003500 MOV #WRHEAD CFMT,E.CS1 ;LOAD EXPECTED CS1
4195 021170 012762 010027 000000 MOV #WRHEAD!CFMT,RKCS1(R2) ;ISSUE COMMAND
4196 021176 012700 000512 MOV #50,*+2,R0 ;ISSUE ENOUGH CLOCKS DATA
4197 ; INDEX PULSE
4198 021202 012762 000440 000026 5$. MOV #DMD.MCLK,RKMR1(R2)
  
```

4199	021210	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4200	021216	005300				DEC	R0	
4201	021220	001370				BNE	5\$	
4202	021222	012700	000004			MOV	#4,R0	;SIMULATE INDEX PULSE
4203	021226	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	
4204	021234	012762	000640	000026	6\$:	MOV	#DMD!MIND.MCLK,RKMR1(R2)	
4205	021242	012762	000240	000026		MOV	#DMD.MIND,RKMR1(R2)	
4206	021250	005300				DEC	R0	
4207	021252	001370				BNE	6\$	
4208	021254	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4209	021262	012737	000300	003510		MOV	#IR!OR,E.CS2	;LOAD EXPECTED CS2
4210	021270	012701	000102			MOV	#66.,R1	;ISSUE 66 NPR TRANSFERS
4211	021274	012700	000050		7\$:	MOV	#40.,R0	
4212	021300	012762	000440	000026	8\$:	MOV	#DMD.MCLK,RKMR1(R2)	
4213	021306	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4214	021314	005300				DEC	R0	
4215	021316	001370				BNE	8\$	
4216	021320	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
4217	021326	016237	000004	003444		MOV	RKBA(R2),T.BA	;STORE BUS ADDRESS REG. 2
4218	021334	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT
4219	021342	022737	064474	003504		CMP	#NPRBUF+2,E.BA	;CHECK IF FIRST WORD
4220	021350	001004				BNE	10\$	;NO, CONTINUE
4221	021352	012700	000024			MOV	#20.,R0	;WAIT FOR OUTPUT READY
4222	021356	005300			9\$:	DEC	R0	
4223	021360	001376				BNE	9\$	
4224	021362	016237	000010	003450	10\$:	MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG. 2
4225	021370	005737	003502			TST	E.WC	;CHECK IF LAST WORD
4226	021374	001003				BNE	11\$	;NO, CONTINUE
4227	021376	012737	000200	003510		MOV	#OR,E.CS2	;LOAD EXPECTED CS2
4228	021404	023737	003500	003440	11\$:	CMP	E.CS1,T.CS1	;CHECK CS1 CORRECT
4229	021412	001401				BEQ	12\$	;YES, CONTINUE
4230	021414	104121				ERROR	121	;CS1 INCORRECT
4231	021416	023737	003510	003450	12\$:	CMP	E.CS2,T.CS2	;CHECK CS2 CORRECT
4232	021424	001401				BEQ	13\$	;YES, CONTINUE
4233	021426	104122				ERROR	122	;CS2 INCORRECT
4234	021430	023737	003504	003444	13\$:	CMP	E.BA,T.BA	;CHECK BUS ADDRESS CORRECT
4235	021436	001401				BEQ	14\$	;YES, CONTINUE
4236	021440	104123				ERROR	123	;BUS ADDRESS INCORRECT
4237	021442	023737	003502	003442	14\$:	CMP	E.WC,T.WC	;CHECK WORD COUNT REG CORRECT
4238	021450	001401				BEQ	15\$	;YES, CONTINUE
4239	021452	104124				ERROR	124	;WORD COUNT INCORRECT
4240	021454	062737	000002	003504	15\$:	ADD	#2,E.BA	;INCREMENT BUS ADDRESS AND
4241	021462	005237	003502			INC	E.WC	;WORD COUNT
4242	021466	005301				DEC	R1	;CHECK IF SILO FULL
4243	021470	001301				BNE	7\$	;NO, GET NEXT WORD
4244	021472	012700	000120			MOV	#2*40.,R0	;ISSUE CLOCKS FOR TWO NPR'S
4245	021476	012762	000440	000026	20\$:	MOV	#DMD.MCLK,RKMR1(R2)	
4246	021504	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
4247	021512	005300				DEC	R0	
4248	021514	001370				BNE	20\$	
4249	021516	162737	000002	003504		SUB	#2,E.BA	;ADJUST BUS ADDRESS AND WORD COUNT
4250	021524	005037	003502			CLR	E.WC	
4251	021530	016237	000000	003440		MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS REG. 1
4252	021536	016237	000010	003450		MOV	RKCS2(R2),T.CS2	;STORE COMMAND AND STATUS REG. 2
4253	021544	016237	000004	003444		MOV	RKBA(R2),T.BA	;STORE BUS ADDRESS
4254	021552	016237	000002	003442		MOV	RKWC(R2),T.WC	;STORE WORD COUNT

```

4255 021560 023737 003500 003440      CMP      E.CS1,T.CS1      :CHECK CS1 CORRECT
4256 021566 001401      BEQ      21$             :YES, CONTINUE
4257 021570 104125      ERROR    125             :CS1 INCORRECT
4258 021572 023737 003510 003450 21$:    CMP      E.CS2,T.CS2      :CHECK CS2 CORRECT
4259 021600 001401      BEQ      22$             :YES, CONTINUE
4260 021602 104126      ERROR    126             :CS2 INCORRECT
4261 021604 023737 003504 003444 22$:    CMP      E.BA,T.BA       :CHECK BUS ADDRESS INCORRECT
4262 021612 001401      BEQ      23$             :YES, CONTINUE
4263 021614 104127      ERROR    127             :BUS ADDRESS INCORRECT
4264 021616 023737 003502 003442 23$:    CMP      E.WC,T.WC       :CHECK WORD COUNT CORRECT
4265 021624 001401      BEQ      24$             :YES, CONTINUE
4266 021626 104130      ERROR    130             :WORD COUNT INCORRECT
4267 021630 012703 064472      MOV      #NPRBUF,R3      :LOAD BUFFER ADDRESS FOR COMPARE
4268 021634 012737 000300 003510      MOV      #IR!OR,E.CS2    :LOAD EXPECTED CS2
4269 021642 012701 000102      MOV      #66.,R1         :LOAD COUNT
4270 021646 005037 003624      CLR      WRDCNT          :INITIALIZE WORD COUNT
4271 021652 016237 000024 003462 25$:    MOV      RKDB(R2),T.DB    :GET DATA BUFFER
4272 021660 012337 003522      MOV      (R3)+,E.DB      :GET EXPECTED DATA
4273 021664 023737 003522 003462      CMP      E.DB,T.DB       :CHECK IF DATA CORRECT
4274 021672 001401      BEQ      26$             :YES, CONTINUE
4275 021674 104131      ERROR    131             :DATA READ INCORRECT
4276 021676 012700 000050      MOV      #40.,R0         :SET STALL
4277 021702 005300      DEC      R0              :RUN STALL TO ZERO
4278 021704 001376      BNF      27$
4279 021706 016237 000000 003440      MOV      RKCS1(R2),T.CS1  :STORE COMMAND AND STATUS REG 1
4280 021714 016237 000010 003450      MOV      RKCS2(R2),T.CS2  :STORE COMMAND AND STATUS REG 2
4281 021722 022701 000001      CMP      #1,R1           :CHECK IF LAST WORD
4282 021726 001003      BNE      28$             :NO, CONTINUE
4283 021730 012737 000100 003510      MOV      #IR,E.CS2       :LOAD EXPECTED CS2
4284 021736 023737 003500 003440 28$:    CMP      E.CS1,T.CS1      :CHECK CS1 CORRECT
4285 021744 001401      BEQ      29$             :YES, CONTINUE
4286 021746 104132      ERROR    132             :CS1 INCORRECT
4287 021750 023737 003510 003450 29$:    CMP      E.CS2,T.CS2      :CHECK CS2 CORRECT
4288 021756 001401      BEQ      30$             :YES, CONTINUE
4289 021760 104133      ERROR    133             :CS2 INCORRECT
4290 021762 005237 003624      INC      WRDCNT          :INCREMENT WORD COUNT
4291 021766 005301      DEC      R1              :CHECK IF FINISHED
4292 021770 001330      BNF      25$             :NO, CONTINUE

```

4293  
 4294  
 4295  
 4296

.SBTTL \*\*MFM READ LOOPBACK TESTS

4297  
4298  
4299  
4300  
4301  
4302  
4303  
4304  
4305  
4306  
4307  
4308  
4309  
4310  
4311  
4312  
4313  
4314  
4315  
4316  
4317  
4318  
4319  
4320  
4321  
4322  
4323  
4324  
4325  
4326  
4327  
4328  
4329  
4330  
4331  
4332  
4333  
4334  
4335  
4336  
4337  
4338  
4339  
4340  
4341  
4342  
4343  
4344  
4345  
4346  
4347  
4348  
4349  
4350  
4351  
4352

021772 000004  
021774 012737 000144 001200  
022002 013702 001270  
022006 012762 100000 000000  
022014 012762 000040 000026  
022022 012762 000025 000000  
022030 012700 000312  
022034 012762 000440 000026  
022042 012762 000040 000026  
022050 005300  
022052 001370  
022054 012762 000140 000026  
022062 012762 000040 000026  
022070 005037 003614  
022074 005037 003616  
022100 012700 000341  
022104 004737 043540  
022110 005300  
022112 001374  
022114 012737 000001 003614  
022122 004737 043540  
022126 012701 000003  
022132 012703 064412  
022136 012737 000025 003500  
022144 012304  
022146 012700 000020  
022152 013737 003614 003616  
022160 006004  
022162 103403  
022164 005037 003614  
022170 000403  
022172 012737 000001 003614  
022200 004737 043540  
022204 016237 000000 003440  
022212 023737 003500 003440  
022220 001417

```
*****  
*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.  
*****  
TST23: SCOPE  
MOV #100.,$TIMES ;DO 100. ITERATIONS  
MOV $BASE,R2 ;LOAD RK611 BASE  
MOV #CCLR,RKCS1(R2) ;CLEAR RK611  
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE  
MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER  
MOV #50.*4+2,R0 ;ISSUE ENOUGH CLCKS UNTIL READY  
1$: MOV #DMD.MCLK,RKMR1(R2) ; FOR SECTOR PULSE  
MOV #DMD,RKMR1(R2)  
DEC R0  
BNE 1$  
MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE  
MOV #DMD,RKMR1(R2)  
CLR PR.BIT ;INITIALIZE PRESENT BIT AND  
CLR M1.BIT ; PREVIOUS BIT  
MOV #225.,R0  
2$: JSR PC,RDBIT ;SIMULATE SYNCH  
DEC R0  
BNE 2$  
MOV #1,PR.BIT  
JSR PC,RDBIT  
MOV #3,R1 ;LOAD NUMBER OF WORDS  
MOV #HEAD1,R3 ;LOAD ADDRESS OF DATA  
MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1  
5$: MOV (R3)+,R4 ;GET DATA  
MOV #16.,R0 ;LOAD BIT COUNT  
6$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT  
ROR R4 ;GET NEXT BIT  
BCS 7$ ;CHECK IF 1  
CLR PR.BIT ;NO, ZERO  
BR 8$ ;SIMULATE READ DATA  
7$: MOV #1,PR.BIT ;ONE  
8$: JSR PC,RDBIT ;SIMULATE READ DATA  
MOV RKCS1(R2),T.CS1 ;READ COMMAND AND STATUS REG. 1  
CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT  
BEQ 9$ ;YES, SIMULATE NEXT BIT  
9$
```

```

4353 022222 012737 000003 003624      MOV    #3,WRDCNT      ;LOAD WORD COUNT
4354 022230 160137 003624      SUB    R1,WRDCNT
4355 022234 012737 000020 003622      MOV    #16,BITCNT    ;LOAD BIT COUNT
4356 022242 160037 003622      SUB    R0,BITCNT
4357 022246 104150      ERROR  150           ;CS1 INCORRECT DURING HEADER
4358 022250 012762 100000 000000      MOV    #CLR,RKCS1(R2) ;CLEAR RK611
4359 022256 000522      BR     TST24         ;;GO ON TO NEXT TEST
4360
4361 022260 005300      9$:    DEC    R0           ;CHECK IF READY FOR NEXT WORD
4362 022262 001333      BNE    6$           ;NO, GET NEXT BIT
4363 022264 005301      DEC    R1           ;CHECK IF HEADER FINISHED
4364 022266 001326      BNE    5$           ;NO, GET NEXT WORD
4365 022270 012700 000004      MOV    #4,R0        ;LOAD COUNT FOR POSTAMBLE
4366 022274 013737 003614 003616 15$:    MOV    PR.BIT,M1.BIT ;STORE LAST BIT
4367 022302 005037 003614      CLR    PR.BIT       ;LOAD NEXT BIT
4368 022306 004737 043540      JSR    PC,RDBIT     ;SIMULATE 1 BIT READ
4369 022312 005300      DEC    R0           ;CHECK IF TIME FOR READY
4370 022314 001367      BNE    15$          ;NO, CONTINUE WITH POSTAMBLE
4371 022316 016237 000000 003440      MOV    RKCS1(R2),T.CS1 ;GET CURRENT CS1
4372 022324 016237 000010 003450      MOV    RKCS2(R2),T.CS2 ;GET CURRENT CS2
4373 022332 012737 000224 003500      MOV    #RDY!RDHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4374 022340 012737 000300 003510      MOV    #OR!IR,E.CS2  ;LOAD EXPECTED CS2
4375 022346 023737 003500 003440      CMP    E.CS1,T.CS1  ;CHECK CS1 CORRECT
4376 022354 001401      BEQ    16$          ;YES, CHECK CS2
4377 022356 104151      ERROR  151           ;CS1 INCORRECT
4378 022360 023737 003510 003450 16$:    CMP    E.CS2,T.CS2  ;CHECK CS2 CORRECT
4379 022366 001401      BEQ    17$          ;YES, CHECK DATA
4380 022370 104152      ERROR  152           ;CS2 INCORRECT
4381 022372 005037 003624      17$:    CLR    WRDCNT       ;INITIALIZE WORD COUNT
4382 022376 012703 064412      MOV    #HEAD1,R3    ;GET ADDRESS OF DATA
4383 022402 012337 003522      20$:    MOV    (R3)+,E.DB   ;GET EXPECTED DATA
4384 022406 016237 000024 003462      MOV    RKDB(R2),T.DB ;GET ACTUAL DATA
4385 022414 016237 000000 003440      MOV    RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4386 022422 016237 000010 003450      MOV    RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4387 022430 022737 000002 003624      CMP    #2,WRDCNT    ;CHECK IF LAST WORD IN DATA BUFFER
4388 022436 001003      BNE    21$          ;NO, CHECK CS1
4389 022440 012737 000100 003510      MOV    #IR,E.CS2    ;STORE EXPECTED CS2
4390 022446 023737 003500 003440 21$:    CMP    E.CS1,T.CS1  ;CHECK CS1 CORRECT
4391 022454 001402      BEQ    22$          ;YES, CHECK CS2
4392 022456 104153      ERROR  153           ;CS1 INCORRECT
4393 022460 000421      BR     TST24         ;;GO ON TO NEXT TEST
4394
4395 022462 023737 003510 003450 22$:    CMP    E.CS2,T.CS2  ;CHECK CS2 CORRECT
4396 022470 001402      BEQ    23$          ;YES, CHECK DATA
4397 022472 104154      ERROR  154           ;CS2 INCORRECT
4398 022474 000413      BR     TST24         ;;GO ON TO NEXT TEST
4399
4400 022476 023737 003522 003462 23$:    CMP    E.DB,T.DB    ;CHECK IF DATA CORRECT
4401 022504 001401      BEQ    24$          ;YES, GET NEXT HEADER WORD
4402 022506 104155      ERROR  155           ;DATA INCORRECT
4403 022510 005237 003624      24$:    INC    WRDCNT       ;INCREMENT WORD COUNT
4404 022514 022737 000003 003624      CMP    #3,WRDCNT    ;CHECK IF ALL THREE WORDS CHECK
4405 022522 001327      BNE    20$          ;NO, GET NEXT WORD

```

```

4406
4407
4408 *****
;*TEST 24 READ LOOPBACK (PART 2)

```

```

4409
4410
4411
4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426 022524 000004
4427 022526 012737 000144 001200
4428 022534 013702 001270
4429 022540 012762 100000 000000
4430 022546 012762 000040 000026
4431 022554 012762 000025 000000
4432 022562 012700 000312
4433 022566 012762 000440 000026 1$:
4434 022574 012762 000040 000026
4435 022602 005300
4436 022604 001370
4437 022606 012762 000140 000026
4438 022614 012762 000040 000026
4439 022622 005037 003614
4440 022626 005037 003616
4441 022632 012700 000341
4442 022636 004737 043540 2$:
4443 022642 005300
4444 022644 001374
4445 022646 012737 000001 003614
4446 022654 004737 043540
4447 022660 012701 000003
4448 022664 012703 064420
4449 022670 012737 000025 003500
4450 022676 012304 5$:
4451 022700 012700 000020
4452 022704 013737 003614 003616 6$:
4453 022712 006004
4454 022714 103403
4455 022716 005037 003614
4456 022722 000403
4457
4458 022724 012737 000001 003614 7$:
4459 022732 004737 043540 8$:
4460 022736 016237 000000 003440
4461 022744 023737 003500 003440
4462 022752 001417
4463 022754 012737 000003 003624
4464 022762 160137 003624

```

\*\*\*\*\*  
 TST24: 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 #RDHEAD,RKCS1(R2) ;:ISSUE READ HEADER  
 MOV #50.\*4+2,R0 ;:ISSUE ENOUGH CLOCKS UNTIL READY  
 MOV #DMD.MCLK,RKMR1(R2) ; FOR SECTOR PULSE  
 MOV #DMD,RKMR1(R2)  
 DEC R0  
 BNE 1\$  
 MOV #DMD!MSP,RKMR1(R2) ;:SIMULATE SECTOR PULSE  
 MOV #DMD,RKMR1(R2)  
 CLR PR.BIT ;:INITIALIZE PRESENT BIT AND  
 CLR M1.BIT ; PREVIOUS BIT  
 MOV #225.,R0  
 JSR PC,RDBIT ;:SIMULATE SYNCH  
 DEC R0  
 BNE 2\$  
 MOV #1,PR.BIT  
 JSR PC,RDBIT  
 MOV #3,R1 ;:LOAD NUMBER OF WORDS  
 MOV #HEAD2,R3 ;:LOAD ADDRESS OF DATA  
 MOV #RDHEAD,E.CS1 ;:LOAD EXPECTED CS1  
 MOV (R3)+,R4 ;:GET DATA  
 MOV #16.,R0 ;:LOAD BIT COUNT  
 MOV PR.BIT,M1.BIT ;:STORE PREVIOUS BIT  
 ROR R4 ;:GET NEXT BIT  
 BCS 7\$ ;:CHECK IF 1  
 CLR PR.BIT ;:NO, ZERO  
 BR 8\$ ;:SIMULATE READ DATA  
 MOV #1,PR.BIT ;:ONE  
 JSR PC,RDBIT ;:SIMULATE READ DATA  
 MOV RKCS1(R2),T.CS1 ;:READ COMMAND AND STATUS REG. 1  
 CMP E.CS1,T.CS1 ;:CHECK IF CS1 CORRECT  
 BEQ 9\$ ;:YES, SIMULATE NEXT BIT  
 MOV #3,WRDCNT ;:LOAD WORD COUNT  
 SUB R1,WRDCNT

```

4465 022766 012737 000020 003622 MOV #16,BITCNT ;LOAD BIT COUNT
4466 022774 160037 003622 SUB R0,BITCNT
4467 023000 104150 ERROR 150 ;CS1 INCORRECT DURING HEADER
4468 023002 012762 100000 000000 MOV #CLR,RKCS1(R2) ;CLEAR RK611
4469 023010 000522 BR TST25 ;;GO ON TO NEXT TEST
4470
4471 023012 005300 9$: DEC R0 ;CHECK IF READY FOR NEXT WORD
4472 023014 001333 BNE 6$ ;NO, GET NEXT BIT
4473 023016 005301 DEC R1 ;CHECK IF HEADER FINISHED
4474 023020 001326 BNE 5$ ;NO, GET NEXT WORD
4475 023022 012700 000004 MOV #4,R0 ;LOAD COUNT FOR POSTAMBLE
4476 023026 013737 003614 003622 15$: MOV PR.BIT,M1.BIT ;STORE LAST BIT
4477 023034 005037 003614 CLR PR.BIT ;LOAD NEXT BIT
4478 023040 004737 043540 JSR PC,RDBIT ;SIMULATE 1 BIT READ
4479 023044 005300 DEC R0 ;CHECK IF TIME FOR READY
4480 023046 001367 BNE 15$ ;NO, CONTINUE WITH POSTAMBLE
4481 023050 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET CURRENT CS1
4482 023056 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;GET CURRENT CS2
4483 023064 012737 000224 003500 MOV #RDY,RDHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4484 023072 012737 000300 003510 MOV #OR!IR,E.CS2 ;LOAD EXPECTED CS2
4485 023100 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4486 023106 001401 BEQ 16$ ;YES, CHECK CS2
4487 023110 104151 ERROR 151 ;CS1 INCORRECT
4488 023112 023737 003510 003450 16$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4489 023120 001401 BEQ 17$ ;YES, CHECK DATA
4490 023122 104152 ERROR 152 ;CS2 INCORRECT
4491 023124 005037 003624 17$: CLR WRDCNT ;INITIALIZE WORD COUNT
4492 023130 012703 064420 MOV #HEAD2,R3 ;GET ADDRESS OF DATA
4493 023134 012337 003522 20$: MOV (R3)+,E.DB ;GET EXPECTED DATA
4494 023140 016237 000024 003462 MOV RKDB(R2),T.DB ;GET ACTUAL DATA
4495 023146 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4496 023154 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4497 023162 022737 000002 003624 CMP #2,WRDCNT ;CHECK IF LAST WORD IN DATA BUFFER
4498 023170 001003 BNE 21$ ;NO, CHECK CS1
4499 023172 012737 000100 003510 MOV #IR,E.CS2 ;STORE EXPECTED CS2
4500 023200 023737 003500 003440 21$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4501 023206 001402 BEQ 22$ ;YES, CHECK CS2
4502 023210 104153 ERROR 153 ;CS1 INCORRECT
4503 023212 000421 BR TST25 ;;GO ON TO NEXT TEST
4504
4505 023214 023737 003510 003450 22$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4506 023222 001402 BEQ 23$ ;YES, CHECK DATA
4507 023224 104154 ERROR 154 ;CS2 INCORRECT
4508 023226 000413 BR TST25 ;;GO ON TO NEXT TEST
4509
4510 023230 023737 003522 003462 23$: CMP E.DB,T.DB ;CHECK IF DATA CORRECT
4511 023236 001401 BEQ 24$ ;YES, GET NEXT HEADER WORD
4512 023240 104155 ERROR 155 ;DATA INCORRECT
4513 023242 005237 003624 24$: INC WRDCNT ;INCREMENT WORD COUNT
4514 023246 022737 000003 003624 CMP #3,WRDCNT ;CHECK IF ALL THREE WORDS CHECK
4515 023254 001327 BNE 20$ ;NO, GET NEXT WORD
4516
4517 *****
4518 ;*TEST 25 READ LOOPBACK (PART 3)
4519 ;*
4520 ;* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER

```



```
4521 :* !N DIAGNOSTIC MOVE. ISSUE A READ HEADER TO AN RK06
4522 :* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
4523 :* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES
4524 :* SIMULATE SECTOR PULSE, 255 ZEROES, A
4525 :* ONE, AND A HEADER CONSISTING OF THE THREE
4526 :* FOLLOWING WORDS:
4527 :*
4528 :*          125252
4529 :*          052525
4530 :*          125252
4531 :*
4532 :* MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
4533 :* IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
4534 :*
4535 :*****
4536 023256 000004 TST25: SCOPE
4537 023260 012737 000144 001200 MOV #100.,$TIMES ;;DO 100. ITERATIONS
4538 023266 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
4539 023272 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4540 023300 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4541 023306 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
4542 023314 012700 000312 MOV #50.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
4543 023320 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4544 023326 012762 000040 000026 MOV #DMD,RKMR1(R2)
4545 023334 005300 DEC R0
4546 023336 001370 BNE 1$
4547 023340 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4548 023346 012762 000040 000026 MOV #DMD,RKMR1(R2)
4549 023354 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT BIT AND
4550 023360 005037 003616 CLR M1.BIT ; PREVIOUS BIT
4551 023364 012700 000341 MOV #225.,R0
4552 023370 004737 043540 2$: JSR PC,RDBIT ;SIMULATE SYNCH
4553 023374 005300 DEC R0
4554 023376 001374 BNE 2$
4555 023400 012737 000001 003614 MOV #1,PR.BIT
4556 023406 004737 043540 JSR PC,RDBIT
4557 023412 012701 000003 MOV #3,R1 ;LOAD NUMBER OF WORDS
4558 023416 012703 064426 MOV #HEAD3,R3 ;LOAD ADDRESS OF DATA
4559 023422 012737 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
4560 023430 012304 5$: MOV (R3)+,R4 ;GET DATA
4561 023432 012700 000020 MOV #16.,R0 ;LOAD BIT COUNT
4562 023436 013737 003614 6$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4563 023444 006004 ROR R4 ;GET NEXT BIT
4564 023446 103403 BCS 7$ ;CHECK IF 1
4565 023450 005037 003614 CLR PR.BIT ;NO, ZERO
4566 023454 000403 BR 8$ ;SIMULATE READ DATA
4567
4568 023456 012737 000001 003614 7$: MOV #1,PR.BIT ;ONE
4569 023464 004737 043540 8$: JSR PC,RDBIT ;SIMULATE READ DATA
4570 023470 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;READ COMMAND AND STATUS REG. 1
4571 023476 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
4572 023504 001417 BEQ 9$ ;YES, SIMULATE NEXT BIT
4573 023506 012737 000003 003624 MOV #3,WRDCNT ;LOAD WORD COUNT
4574 023514 160137 003624 SUB R1,WRDCNT
4575 023520 012737 000020 003622 MOV #16.,BITCNT ;LOAD BIT COUNT
4576 023526 160037 003622 SUB R0,BITCNT
```

```

4577 023532 104150          ERROR 150          ;CS1 INCORRECT DURING HEADER
4578 023534 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4579 023542 000522          BR TST26          ;;GO ON TO NEXT TEST
4580
4581 023544 005300          9$: DEC R0          ;CHECK IF READY FOR NEXT WORD
4582 023546 001333          BNE 6$          ;NO, GET NEXT BIT
4583 023550 005301          DEC R1          ;CHECK IF HEADER FINISHED
4584 023552 001326          BNE 5$          ;NO, GET NEXT WORD
4585 023554 012700 000004          MOV #4,R0       ;LOAD COUNT FOR POSTAMBLE
4586 023560 013737 003614 003616 15$: MOV PR.BIT,M1.BIT ;STORE LAST BIT
4587 023566 005037 003614          CLR PR.BIT      ;LOAD NEXT BIT
4588 023572 004737 043540          JSR PC,RDBIT    ;SIMULATE 1 BIT READ
4589 023576 005300          DEC R0          ;CHECK IF TIME FOR READY
4590 023600 001367          BNE 15$        ;NO, CONTINUE WITH POSTAMBLE
4591 023602 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET CURRENT CS1
4592 023610 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;GET CURRENT CS2
4593 023616 012737 000224 003500 MOV #RDY!RDHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4594 023624 012737 000300 003510 MOV #OR!IR,E.CS2  ;LOAD EXPECTED CS2
4595 023632 023737 003500 003440 CMP E.CS1,T.CS1  ;CHECK CS1 CORRECT
4596 023640 001401          BEQ 16$        ;YES, CHECK CS2
4597 023642 104151          ERROR 151      ;CS1 INCORRECT
4598 023644 023737 003510 003450 16$: CMP E.CS2,T.CS2  ;CHECK CS2 CORRECT
4599 023652 001401          BEQ 17$        ;YES, CHECK DATA
4600 023654 104152          ERROR 152      ;CS2 INCORRECT
4601 023656 005037 003624          CLR WRDCNT     ;INITIALIZE WORD COUNT
4602 023662 012703 064426          MOV #HEAD3,R3  ;GET ADDRESS OF DATA
4603 023666 012337 003522          MOV (R3)+,E.DB ;GET EXPECTED DATA
4604 023672 016237 000024 003462 MOV RKDB(R2),T.DB ;GET ACTUAL DATA
4605 023700 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4606 023706 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4607 023714 022737 000002 003624 CMP #2,WRDCNT   ;CHECK IF LAST WORD IN DATA BUFFER
4608 023722 001003          BNE 21$        ;NO, CHECK CS1
4609 023724 012737 000100 003510 MOV #IR,E.CS2   ;STORE EXPECTED CS2
4610 023732 023737 003500 003440 21$: CMP E.CS1,T.CS1  ;CHECK CS1 CORRECT
4611 023740 001402          BEQ 22$        ;YES, CHECK CS2
4612 023742 104153          ERROR 153      ;CS1 INCORRECT
4613 023744 000421          BR TST26      ;;GO ON TO NEXT TEST
4614
4615 023746 023737 003510 003450 22$: CMP E.CS2,T.CS2  ;CHECK CS2 CORRECT
4616 023754 001402          BEQ 23$        ;YES, CHECK DATA
4617 023756 104154          ERROR 154      ;CS2 INCORRECT
4618 023760 000413          BR TST26      ;;GO ON TO NEXT TEST
4619
4620 023762 023737 003522 003462 23$: CMP E.DB,T.DB   ;CHECK IF DATA CORRECT
4621 023770 001401          BEQ 24$        ;YES, GET NEXT HEADER WORD
4622 023772 104155          ERROR 155      ;DATA INCORRECT
4623 023774 005237 003624          INC WRDCNT     ;INCREMENT WORD COUNT
4624 024000 022737 000003 003624 24$: CMP #3,WRDCNT  ;CHECK IF ALL THREE WORDS CHECK
4625 024006 001327          BNE 20$        ;NO, GET NEXT WORD
4626
4627 *****
4628 ;*TEST 26 READ LOOPBACK (PART 4)
4629 ;*
4630 ;* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
4631 ;* IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN RK06
4632 ;* IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.

```

```

4633
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646 024010 000004
4647 024012 012737 000144 001200
4648 024020 013702 001270
4649 024024 012762 100000 000000
4650 024032 012762 000040 000026
4651 024040 012762 000025 000000
4652 024046 012700 000312
4653 024052 012762 000440 000026 1$:
4654 024060 012762 000040 000026
4655 024066 005300
4656 024070 001370
4657 024072 012762 000140 000026
4658 024100 012762 000040 000026
4659 024106 005037 003614
4660 024112 005037 003616
4661 024116 012700 000341
4662 024122 004737 043540 2$:
4663 024126 005300
4664 024130 001374
4665 024132 012737 000001 003614
4666 024140 004737 043540
4667 024144 012701 000003
4668 024150 012703 064442
4669 024154 012737 000025 003500
4670 024162 012304 5$:
4671 024164 012700 000020
4672 024170 013737 003614 003616 6$:
4673 024176 006004
4674 024200 103403
4675 024202 005037 003614
4676 024206 000403
4677
4678 024210 012737 000001 003614 7$:
4679 024216 004737 043540 8$:
4680 024222 016237 000000 003440
4681 024230 023757 003500 003440
4682 024236 001417
4683 024240 012737 000003 003624
4684 024246 160137 003624
4685 024252 012737 000020 003622
4686 024260 160037 003622
4687 024264 104150
4688 024266 012762 100000 000000

```

```

: *
: * 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
MOV #100.,$TIMES ;;DO 100. ITERATIONS
MOV $BASE,R2 ;LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;CLEAR RK611
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
MOV #50.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
1$: MOV #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
MOV #DMD,RKMR1(R2)
DEC R0
BNF 1$
MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
MOV #DMD,RKMR1(R2)
CLR PR.BIT ;INITIALIZE PRESENT BIT AND
CLR M1.BIT ; PREVIOUS BIT
MOV #225.,R0
2$: JSR PC,RDBIT ;SIMULATE SYNCH
DEC R0
BNE 2$
MOV #1,PR.BIT
JSR PC,RDBIT
MOV #3,R1 ;LOAD NUMBER OF WORDS
MOV #HEAD4,R3 ;LOAD ADDRESS OF DATA
MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
5$: MOV (R3)+,R4 ;GET DATA
MOV #16.,R0 ;LOAD BIT COUNT
6$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
ROR R4 ;GET NEXT BIT
BCS 7$ ;CHECK IF 1
CLR PR.BIT ;NO, ZERO
BR 8$ ;SIMULATE READ DATA
7$: MOV #1,PR.BIT ;ONE
8$: JSR PC,RDBIT ;SIMULATE READ DATA
MOV RKCS1(R2),T.CS1 ;READ COMMAND AND STATUS REG.
CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
BEQ 9$ ;YES, SIMULATE NEXT BIT
MOV #3,WRDCNT ;LOAD WORD COUNT
SUB R1,WRDCNT
MOV #16.,BITCNT ;LOAD BIT COUNT
SUB R0,BITCNT
ERROR 150 ;CS1 INCORRECT DURING HEADER
MOV #CCLR,RKCS1(R2) ;CLEAR RK611

```

```

4689 024274 000522 BR TST27 ;;GO ON TO NEXT TEST
4690
4691 024276 005300 9$: DEC R0 ;CHECK IF READY FOR NEXT WORD
4692 024300 001333 BNE 6$ ;NO, GET NEXT BIT
4693 024302 005301 DEC R1 ;CHECK IF HEADER FINISHED
4694 024304 001326 BNE 5$ ;NO, GET NEXT WORD
4695 024306 012700 000004 MOV #4,R0 ;LOAD COUNT FOR POSTAMBLE
4696 024312 013737 003614 003616 15$: MOV PR.BIT,M1.BIT ;STORE LAST BIT
4697 024320 005037 003614 CLR PR.BIT ;LOAD NEXT BIT
4698 024324 004737 043540 JSR PC,RDBIT ;SIMULATE 1 BIT READ
4699 024330 005300 DEC R0 ;CHECK IF TIME FOR READY
4700 024332 001367 BNE 15$ ;NO, CONTINUE WITH POSTAMBLE
4701 024334 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET CURRENT CS1
4702 024342 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;GET CURRENT CS2
4703 024350 012737 000224 003500 MOV #RDY!RDHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4704 024356 012737 000300 003510 MOV #OR.IR,E.CS2 ;LOAD EXPECTED CS2
4705 024364 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4706 024372 001401 BEQ 16$ ;YES, CHECK CS2
4707 024374 104151 ERROR 151 ;CS1 INCORRECT
4708 024376 023737 003510 003450 16$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4709 024404 001401 BFG 17$ ;YES, CHECK DATA
4710 024406 104152 ERROR 152 ;CS2 INCORRECT
4711 024410 005037 003624 17$: CLR WRDCNT ;INITIALIZE WORD COUNT
4712 024414 012703 064442 MOV #HEAD4,R3 ;GET ADDRESS OF DATA
4713 024420 012337 003522 20$: MOV (R3)+,E.DB ;GET EXPECTED DATA
4714 024424 016237 000024 003462 MOV RKDB(R2),T.DB ;GET ACTUAL DATA
4715 024432 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4716 024440 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4717 024446 022737 000002 003624 CMP #2,WRDCNT ;CHECK IF LAST WORD IN DATA BUFFER
4718 024454 001003 BNE 21$ ;NO, CHECK CS1
4719 024456 012737 000100 003510 MOV #IR,E.CS2 ;STORE EXPECTED CS2
4720 024464 023737 003500 003440 21$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4721 024472 001402 BEQ 22$ ;YES, CHECK CS2
4722 024474 104153 ERROR 153 ;CS1 INCORRECT
4723 024476 000421 BR TST27 ;;GO ON TO NEXT TEST
4724
4725 024500 023737 003510 003450 22$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4726 024506 001402 BEQ 23$ ;YES, CHECK DATA
4727 024510 104154 ERROR 154 ;CS2 INCORRECT
4728 024512 000413 BR TST27 ;;GO ON TO NEXT TEST
4729
4730 024514 023737 003522 003462 23$: CMP E.DB,T.DB ;CHECK IF DATA CORRECT
4731 024522 001401 BEQ 24$ ;YES, GET NEXT HEADER WORD
4732 024524 104155 ERROR 155 ;DATA INCORRECT
4733 024526 005237 003624 24$: INC WRDCNT ;INCREMENT WORD COUNT
4734 024532 022737 000003 003624 CMP #3,WRDCNT ;CHECK IF ALL THREE WORDS CHECK
4735 024540 001327 BNE 20$ ;NO, GET NEXT WORD

```

```

*****
*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

```

4736  
4737  
4738  
4739  
4740  
4741  
4742  
4743  
4744

```
4745 : * ONE, AND A HEADER CONSISTING OF THE THREE
4746 : * FOLLOWING WORDS.
4747 : *
4748 : * 052012
4749 : * 100520
4750 : * 052012
4751 : *
4752 : * MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
4753 : * IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
4754 : *
4755 : *
4756 : *****
4756 024542 000004 TST27: SCOPE
4757 024544 012737 000144 001200 MOV #100.,$TIMES ;;DO 100. ITERATIONS
4758 024552 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
4759 024556 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4760 024564 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4761 024572 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;ISSUE READ HEADER
4762 024600 012700 000312 MOV #50.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
4763 024604 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2) ; FOR SFCTOR PULSE
4764 024612 012762 000040 000026 MOV #DMD,RKMR1(R2)
4765 024620 005300 DFC R0
4766 024622 001370 BNE 1$
4767 024624 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4768 024632 012762 000040 000026 MOV #DMD,RKMR1(R2)
4769 024640 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT BIT AND
4770 024644 005037 003616 CLR M1.BIT ; PREVIOUS BIT
4771 024650 012700 000341 MOV #225.,R0
4772 024654 004737 043540 2$: JSR PC,RDBIT ;SIMULATE SYNCH
4773 024660 005300 DEC R0
4774 024662 001374 BNE 2$
4775 024664 012737 000001 003614 MOV #1,PR.BIT
4776 024672 004737 043540 JSR PC,RDBIT
4777 024676 012701 000003 MOV #3,R1 ;LOAD NUMBER OF WORDS
4778 024702 012703 064450 MOV #HEAD5,R3 ;LOAD ADDRESS OF DATA
4779 024706 012707 000025 003500 MOV #RDHEAD,E.CS1 ;LOAD EXPECTED CS1
4780 024714 012304 5$: MOV (R3)+,R4 ;GET DATA
4781 024716 012700 000020 MOV #16.,R0 ;LOAD BIT COUNT
4782 024722 013737 003614 6$: MOV PR.BIT,M1.BIT ;STORE PREVIOUS BIT
4783 024730 006004 ROR R4 ;GET NEXT BIT
4784 024732 103403 BCS 7$ ;CHECK IF 1
4785 024734 005037 003614 CLR PR.BIT ;NO, ZERO
4786 024740 000403 BR 8$ ;SIMULATE READ DATA
4787
4788 024742 012737 000001 003614 7$: MOV #1,PR.BIT ;ONE
4789 024750 004737 043540 8$: JSR PC,RDBIT ;SIMULATE READ DATA
4790 024754 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;READ COMMAND AND STATUS REG. 1
4791 024762 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
4792 024770 001417 BEQ 9$ ;YES, SIMULATE NEXT BIT
4793 024772 012737 000003 003624 MOV #3,WRDCNT ;LOAD WORD COUNT
4794 025000 160137 003624 SUB R1,WRDCNT
4795 025004 012737 000020 003622 MOV #16.,BITCNT ;LOAD BIT COUNT
4796 025012 160037 003622 SUB R0,BITCNT
4797 025016 104150 ERROR 150 ;CS1 INCORRECT DURING HEADER
4798 025020 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4799 025026 000522 BR TST30 ;GO ON TO NEXT TEST
4800
```

```

4801 025030 005300          9$: DEC R0 ;CHECK IF READY FOR NEXT WORD
4802 025032 001333          BNE 6$ ;NO, GET NEXT BIT
4803 025034 005301          DEC R1 ;CHECK IF HEADER FINISHED
4804 025036 001326          BNE 5$ ;NO, GET NEXT WORD
4805 025040 012700 000004    MOV #4,R0 ;LOAD COUNT FOR POSTAMBLE
4806 025044 013737 003614 003616 15$: MOV PR.BIT,M1.BIT ;STORE LAST BIT
4807 025052 005037 003614    CLR PR.BIT ;LOAD NEXT BIT
4808 025056 004737 043540    JSR PC,RDBIT ;SIMULATE 1 BIT READ
4809 025062 005300          DEC R0 ;CHECK IF TIME FOR READY
4810 025064 001367          BNE 15$ ;NO, CONTINUE WITH POSTAMBLE
4811 025066 016237 000000 003440  MOV RKCS1(R2),T.CS1 ;GET CURRENT CS1
4812 025074 016237 000010 003450  MOV RKCS2(R2),T.CS2 ;GET CURRENT CS2
4813 025102 012737 000224 003500  MOV #RDY!RDHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
4814 025110 012737 000300 003510  MOV #OR!IR,E.CS2 ;LOAD EXPECTED CS2
4815 025116 023737 003500 003440  CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4816 025124 001401          BEQ 16$ ;YES, CHECK CS2
4817 025126 104151          ERROR 151 ;CS1 INCORRECT
4818 025130 023737 003510 003450 16$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4819 025136 001401          BEQ 17$ ;YES, CHECK DATA
4820 025140 104152          ERROR 152 ;CS2 INCORRECT
4821 025142 005037 003624          CLR WRDCNT ;INITIALIZE WORD COUNT
4822 025146 012703 064450          MOV #HEAD5,R3 ;GET ADDRESS OF DATA
4823 025152 012337 003522          MOV (R3)+,E.DB ;GET EXPECTED DATA
4824 025156 016237 000024 003462  MOV RKDB(R2),T.DB ;GET ACTUAL DATA
4825 025164 016237 000000 003440  MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
4826 025172 016237 000010 003450  MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
4827 025200 022737 000002 003624  CMP #2,WRDCNT ;CHECK IF LAST WORD IN DATA BUFFER
4828 025206 001003          BNE 21$ ;NO, CHECK CS1
4829 025210 012737 000100 003510  MOV #IR,E.CS2 ;STORE EXPECTED CS2
4830 025216 023737 003500 003440 21$: CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
4831 025224 001402          BEQ 22$ ;YES, CHECK CS2
4832 025226 104153          ERROR 153 ;CS1 INCORRECT
4833 025230 000421          BR TST30 ;GO ON TO NEXT TEST
4834
4835 025232 023737 003510 003450 22$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
4836 025240 001402          BEQ 23$ ;YES, CHECK DATA
4837 025242 104154          ERROR 154 ;CS2 INCORRECT
4838 025244 000413          BR TST30 ;GO ON TO NEXT TEST
4839
4840 025246 023737 003522 003462 23$: CMP E.DB,T.DB ;CHECK IF DATA CORRECT
4841 025254 001401          BEQ 24$ ;YES, GET NEXT HEADER WORD
4842 025256 104155          ERROR 155 ;DATA INCORRECT
4843 025260 005237 003624          INC WRDCNT ;INCREMENT WORD COUNT
4844 025264 022737 000003 003624 24$: CMP #3,WRDCNT ;CHECK IF ALL THREE WORDS CHECK
4845 025272 001327          BNE 20$ ;NO, GET NEXT WORD

```

```

4846
4847 ;*****
4848 ;*TEST 30 READ HEADER IN 18 BIT MODE
4849 ;*
4850 ;* CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
4851 ;* IN 24 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
4852 ;* CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
4853 ;* SIMULATE SECTOR PULSE, 255 ZEROES, A
4854 ;* ONE, AND A HEADER CONSISTING OF THE THREE
4855 ;* FOLLOWING WORDS:
4856 ;*

```

```
4857          : *          177777
4858          : *          000000
4859          : *          177777
4860          : *
4861          : *          MAKE SURE THAT READY COMES UP AFTER THE THIRD WORD
4862          : *          IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
4863          : *
4864          : *
4865          : *****
4865 025274 000004 TST30: SCOPE
4866 025276 012737 000144 001200 MOV #100.,$TIMES ;;DO 100. ITERATIONS
4867 025304 013702 001270 MOV $BASE,R2 ;LOAD RK611
4868 025310 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4869 025316 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
4870 025324 012762 010025 000000 MOV #CFMT!RDHEAD,RKCS1(R2) ;ISSUE READ HEADER (24 SECTOR FORMAT)
4871 025332 012700 000312 MOV #50.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL READY
4872 025336 012762 000440 000026 1$: MOV #DMD.MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4873 025344 012762 000040 000026 MOV #DMD,RKMR1(R2)
4874 025352 005300 DEC R0
4875 025354 001370 BNE 1$
4876 025356 012762 000140 000026 MOV #DMD.MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
4877 025364 012762 000040 000026 MOV #DMD,RKMR1(R2)
4878 025372 005037 003614 CLR PR.BIT ;INITIALIZE PRESENT BIT AND
4879 025376 005037 003616 CLR M1.BIT ; PREVIOUS BIT
4880 025402 012700 000377 MOV #255.,R0
4881 025406 004737 043540 2$: JSR PC,RDBIT ;SIMULATE SYNCH
4882 025412 005300 DEC R0
4883 025414 001374 BNE 2$
4884 025416 012737 000001 003614 MOV #1,PR.BIT
4885 025424 004737 043540 JSR PC,RDBIT
4886 025430 012701 000003 MOV #3,R1 ;LOAD NUMBER OF WORDS
4887 025434 012703 064412 MOV #HEAD1,R3 ;LOAD ADDRESS OF DATA
4888 025440 012737 010025 003500 MOV #CFMT.RDHEAD,E.CS1 ;LOAD EXPECTED CS1
4889 025446 012304 5$: MOV (R3)+,R4 ;GET DATA
4890 025450 012700 000020 MOV #16.,R0 ;LOAD BIT COUNT
4891 025454 013737 003614 6$: MOV PR.BIT,M1.BIT ;STORE PRESENT BIT
4892 025462 006004 ROR R4 ;GET NEXT BIT
4893 025464 103403 BCS 7$ ;CHECK IF 1
4894 025466 005037 003614 CLR PR.BIT ;NO, ZERO
4895 025472 000403 BR 8$ ;SIMULATE READ DATA
4896
4897 025474 012737 000001 003614 7$: MOV #1,PR.BIT ;ONE
4898 025502 004737 043540 8$: JSR PC,RDBIT ;SIMULATE READ DATA
4899 025506 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;READ COMMAND AND STATUS REG. 1
4900 025514 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
4901 025522 001417 BEQ 9$ ;YES, SIMULATE NEXT BIT
4902 025524 012737 000003 003624 MOV #3,WRDCNT ;LOAD WORD COUNT
4903 025532 160137 003624 SUB R1,WRDCNT
4904 025536 012737 000020 003622 MOV #16.,BITCNT ;LOAD BIT COUNT
4905 025544 160037 003622 SUB R0,BITCNT
4906 025550 104156 ERROR 156 ;CS1 INCORRECT DURING HEADER
4907 025552 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
4908 025560 000522 BR TST31 ;GO ON TO NEXT TEST
4909
4910 025562 005300 9$: DEC R0 ;CHECK IF READY FOR NEXT WORD
4911 025564 001333 BNE 6$ ;NO, GET NEXT BIT
4912 025566 005301 DEC R1 ;CHECK IF HEADER FINISHED
```

```

4913 025570 001326          BNE      5$          :NO, GET NEXT WORD
4914 025572 012700 000004    MOV      #4,R0       :LOAD COUNT FOR POSTAMBLE
4915 025576 013737 003614 003616 15$:  MOV      PR.BIT,M1.BIT :STORE LAST BIT
4916 025604 005037 003614    CLR      PR.BIT      :LOAD NEXT BIT
4917 025610 004737 043540    JSR      PC,RDBIT    :READ BIT
4918 025614 005300          DEC      R0          :CHECK IF TIME FOR READY
4919 025616 001367          BNE      15$         :NO, CONTINUE WITH POSTAMBLE
4920 025620 016237 000000 003440    MOV      RKCS1(R2),T.CS1 :GET CURRENT CS1
4921 025626 016237 000010 003450    MOV      RKCS2(R2),T.CS2 :GET CURRENT CS2
4922 025634 012737 010224 003500    MOV      #CFMT.RDY!RDHEAD<^C<GO>>,E.CS1 :LOAD EXPECTED CS1
4923 025642 012737 000300 003510    MOV      #OR!IR,E.CS2   :LOAD EXPECTED CS2
4924 025650 023737 003500 003440    CMP      E.CS1,T.CS1    :CHECK CS1 CORRECT
4925 025656 001401          BEQ      16$         :YES, CHECK CS2
4926 025660 104157          ERROR    157         :CS1 INCORRECT
4927 025662 023737 003510 003450 16$:  CMP      E.CS2,T.CS2    :CHECK CS2 CORRECT
4928 025670 001401          BEQ      17$         :YES, CHECK DATA
4929 025672 104160          ERROR    160         :CS2 INCORRECT
4930 025674 005037 003624          CLR      WRDCNT       :INITIALIZE WORD COUNT
4931 025700 012703 064412    MOV      #HEAD1,R3     :GET ADDRESS OF DATA
4932 025704 012337 003522          MOV      (R3)+,E.DB    :GET EXPECTED DATA
4933 025710 016237 000024 003462    MOV      RKDB(R2),T.DB  :GET ACTUAL DATA
4934 025716 016237 000000 003440    MOV      RKCS1(R2),T.CS1 :STORE COMMAND AND STATUS REG. 1
4935 025724 016237 000010 003450    MOV      RKCS2(R2),T.CS2 :STORE COMMAND AND STATUS REG. 2
4936 025732 022737 000002 003624    CMP      #2,WRDCNT     :CHECK IF LAST WORD IN DATA BUFFER
4937 025740 001003          BNE      21$         :NO, CHECK CS1
4938 025742 012737 000100 003510    MOV      #IR,E.CS2     :LOAD EXPECTED CS2
4939 025750 023737 003500 003440 21$:  CMP      E.CS1,T.CS1    :CHECK CS1 CORRECT
4940 025756 001402          BEQ      22$         :YES, CHECK CS2
4941 025760 104161          ERROR    161         :CS1 INCORRECT
4942 025762 000421          BR       TST31       ;;GO ON TO NEXT TEST
4943
4944 025764 023737 003510 003450 22$:  CMP      E.CS2,T.CS2    :CHECK CS2 CORRECT
4945 025772 001402          BEQ      23$         :YES, CHECK DATA BUFFER
4946 025774 104162          ERROR    162         :CS2 INCORRECT
4947 025776 000413          BR       TST31       ;;GO ON TO NEXT TEST
4948
4949 026000 023737 003522 003462 23$:  CMP      E.DB,T.DB     :CHECK DATA BUFFER CORRECT
4950 026006 001401          BEQ      24$         :YES, GET NEXT WORD
4951 026010 104163          ERROR    163         :DATA BUFFER INCORRECT
4952 026012 005237 003624          INC      WRDCNT       :INCREMENT WORD COUNT
4953 026016 022737 000003 003624 24$:  CMP      #3,WRDCNT     :CHECK IF FINISHED
4954 026024 001327          BNE      20$         :NO READ NEXT WORD
4955

```

```

*****
*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 C.
*          CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES.
*          SIMULATE SECTOR PULSE AND 350 ZEROES.  MAKE
*          SURE READY REMAINS RESET AND THE SILO REMAINS
*          EMPTY.
*****
TST31: SCOPE

```

4968 026026 000004



```

4969 026030 012737 000144 001200      MOV      #100.,$TIMES      ;;DO 100. ITERATIONS
4970 026036 013702 001270      MOV      $BASE,R2        ;;LOAD RK611 BASE
4971 026042 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;;CLEAR RK611
4972 026050 012762 000040 000026      MOV      #DMD,RKMR1(R2)  ;;PUT RK611 IN MAINT MODE
4973 026056 012762 000025 000000      MOV      #RDHEAD,RKCS1(R2) ;;ISSUE READ HEAD
4974 026064 012700 000312      MOV      #50.*4+2,R0     ;;ISSUE ENOUGH CLOCKS UNTIL READY
4975 026070 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
4976 026076 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4977 026104 005300      DEC      R0
4978 026106 001370      BNE     1$
4979 026110 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE TO SECTOR PULSE
4980 026116 012762 000040 000026      MOV      #DMD,RKMR1(R2)
4981 026124 005037 003614      CLR     PR.BIT          ;;INITIALIZE PRESENT AND
4982 026130 005037 003616      CLR     M1.BIT          ;; PREVIOUS BIT
4983 026134 012737 000025 003500      MOV      #RDHEAD,E.CS1   ;;LOAD EXPECTED CS1
4984 026142 012737 000100 003510      MOV      #IR,E.CS2      ;;LOAD EXPECTED CS2
4985 026150 005037 003622      CLR     BITCNT          ;;SIMULATE 350 ZEROES
4986 026154 004737 043540 2$:      JSR     PC,RDBIT
4987 026160 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;;STORE CS1
4988 026166 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;;STORE CS2
4989 026174 023737 003500 003440      CMP     E.CS1,T.CS1     ;;CHECK IF CS1 CORRECT
4990 026202 001402      BEQ     3$              ;;YES, CHECK CS2
4991 026204 104164      ERROR   164            ;;CS1 INCORRECT
4992 026206 000447      BR      TST32          ;;GO ON TO NEXT TEST
4993
4994 026210 023737 003510 003450 3$:      CMP     E.CS2,T.CS2     ;;CHECK IF CS2 CORRECT
4995 026216 001402      BEQ     4$              ;;YES, CHECK IF SILO EMPTY
4996 026220 104165      ERROR   165            ;;CS2 INCORRECT
4997 026222 000441      BR      TST32          ;;GO ON TO NEXT TEST
4998 026224 005237 003622 4$:      INC     BITCNT          ;;INCREMENT BIT COUNT
4999 026230 022737 000536 003622      CMP     #350.,BITCNT    ;;CHECK IF FINISHED
5000 026236 001346      BNE     2$              ;;NO, SIMULATE NEXT ZERO
5001 026240 005762 000024      TST     RKDB(R2)        ;;READ DATA BUFFER
5002 026244 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;;STORE CS1 AND CS2
5003 026252 016237 000010 003450      MOV      RKCS2(R2),T.CS2
5004 026260 012737 100224 003500      MOV      #CERR!RDY!RDHEAD<^C<GO>>,E.CS1 ;;LOAD EXPECT CS1
5005 026266 012737 100100 003510      MOV      #DCK IR,E.CS2  ;;LOAD EXPECTED CS2
5006 026274 023737 003500 003440      CMP     E.CS1,T.CS1     ;;CHECK FOR CONTROLLER ERROR
5007 026302 001401      BEQ     5$              ;;YES, CHECK FOR DATA LATE
5008 026304 104166      ERROR   166            ;;CS1 INCORRECT
5009 026306 023737 003510 003450 5$:      CMP     E.CS2,T.CS2     ;;CHECK FOR DATA LATE
5010 026314 001401      BEQ     6$              ;;YES, CHECK RK611
5011 026316 104167      ERROR   167            ;;CS2 INCORRECT
5012 026320 012762 100000 000000 6$:      MOV      #CCLR,RKCS1(R2) ;;CLEAR RK611
5013

```

```

5014 *****
5015 *TEST 32      ZERO SYNCH ON READ
5016 *
5017 *
5018 *      CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
5019 *      IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
5020 *      CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES,
5021 *      SIMULATE SECTOR PULSE, 255 ZEROES SHIFTED BY A HALF
5022 *      BIT TIME, A ONE, AND A HEADER CONSISTING OF THE
5023 *      THREE FOLLOWING WORDS:
5024 *
5025 *      177777

```

```
5025          :*          000000
5026          :*          177777
5027          :*
5028          :*          MAKE SURE THAT READY COMES AFTER THE THIRD WORD
5029          :*          IS TRANSFERRED. CHECK THE SILO FOR CORRECT CONTENTS.
5030          :*
5031          :*
5032          :*****
5032 026326 000004 TST32: SCOPE
5033 026330 012737 000144 001200 MOV #100.,$TIMES ;:DO 100. ITERATIONS
5034 026336 013702 001270 MOV $BASE,R2 ;:LOAD RK611 BASE
5035 026342 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
5036 026350 012762 000040 000026 MOV #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
5037 026356 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;:ISSUE READ HEADER
5038 026364 012700 000312 MOV #50.*4+2,R0 ;:ISSUE ENOUGH CLOCKS UNTIL READY
5039 026370 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2) ; FOR SECTOR PULSE
5040 026376 012762 000040 000026 MOV #DMD,RKMR1(R2)
5041 026404 005300 DEC R0
5042 026406 001370 BNE 1$
5043 026410 012762 000140 000026 MOV #DMD!MSP,RKMR1(R2) ;:SIMULATE SECTOR PULSE
5044 026416 012762 000040 000026 MOV #DMD,RKMR1(R2)
5045 026424 012762 000440 000026 MOV #DMD!MCLK,RKMR1(R2) ;:SHIFT DATA ONE HALF BIT TIME
5046 026432 012762 000040 000026 MOV #DMD,RKMR1(R2)
5047 026440 005037 003614 CLR PR.BIT ;:INITIALIZE PRESENT BIT AND
5048 026444 005037 003616 CLR M1.BIT ; PREVIOUS BIT
5049 026450 012700 000341 MOV #225.,R0
5050 026454 004737 043540 2$: JSR PC,RDBIT ;:SIMULATE SYNCH
5051 026460 005300 DEC R0
5052 026462 001374 BNE 2$
5053 026464 012737 000001 003614 MOV #1,PR.BIT
5054 026472 004737 043540 JSR PC,RDBIT
5055 026476 012701 000003 MOV #3,R1 ;:LOAD NUMBER OF WORDS
5056 026502 012703 064412 MOV #HEAD1,R3 ;:LOAD ADDRESS OF DATA
5057 026506 012737 000025 003500 MOV #RDHEAD,E.CS1 ;:LOAD EXPECTED CS1
5058 026514 012304 5$: MOV (R3)+,R4 ;:GET DATA
5059 026516 012700 000020 MOV #16.,R0 ;:LOAD BIT COUNT
5060 026522 013737 003614 003616 6$: MOV PR.BIT,M1.BIT ;:STORE PREVIOUS BIT
5061 026530 006004 ROR R4 ;:GET NEXT BIT
5062 026532 103403 BCS 7$ ;:CHECK IF 1
5063 026534 005037 003614 CLR PR.BIT ;:NO, ZERO
5064 026540 000403 BR 8$ ;:SIMULATE READ DATA
5065
5066 026542 012737 000001 003614 7$: MOV #1,PR.BIT ;:ONE
5067 026550 004737 043540 8$: JSR PC,RDBIT ;:SIMULATE READ DATA
5068 026554 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;:READ COMMAND AND STATUS REG. 1
5069 026562 023737 003500 003440 CMP E.CS1,T.CS1 ;:CHECK IF CS1 CORRECT
5070 026570 001417 BEQ 9$ ;:YES, SIMULATE NEXT BIT
5071 026572 012737 000003 003624 MOV #3,WRDCNT ;:LOAD WORD COUNT
5072 026600 160137 003624 SUB R1,WRDCNT
5073 026604 012737 000020 003622 MOV #16.,BITCNT ;:LOAD BIT COUNT
5074 026612 160037 003622 SUB R0,BITCNT
5075 026616 104150 ERROR 150 ;:CS1 INCORRECT DURING HEADER
5076 026620 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
5077 026626 000522 BR TST33 ;:GO ON TO NEXT TEST
5078
5079 026630 005300 9$: DEC R0 ;:CHECK IF READY FOR NEXT WORD
5080 026632 001333 BNE 6$ ;:NO, GET NEXT BIT
```

```

5081 026634 005301          DEC      R1          ;CHECK IF HEADER FINISHED
5082 026636 001326          BNE      5$          ;NO, GET NEXT WORD
5083 026640 012700 000004      MOV      #4,R0       ;LOAD COUNT FOR POSTAMBLE
5084 026644 013737 003614 003616 15$:  MOV      PR,BIT,M1.BIT ;STORE LAST BIT
5085 026652 005037 003614          CLR      PR,BIT     ;LOAD NEXT BIT
5086 026656 004737 043540          JSR      PC,RDBIT   ;SIMULATE 1 BIT READ
5087 026662 005300          DEC      R0         ;CHECK IF TIME FOR READY
5088 026664 001367          BNE      15$        ;NO, CONTINUE WITH POSTAMBLE
5089 026666 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;GET CURRENT CS1
5090 026674 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;GET CURRENT CS2
5091 026702 012737 000224 003500      MOV      #RDY!RDHEAD<<^C<<GO>>,E.CS1 ;LOAD EXPECTED CS1
5092 026710 012737 000300 003500      MOV      #OR!IR,E.CS2  ;LOAD EXPECTED CS2
5093 026716 023737 003500 003440      CMP      E.CS1,T.CS1  ;CHECK CS1 CORRECT
5094 026724 001401          BEQ      16$        ;YES, CHECK CS2
5095 026726 104151          ERROR   151        ;CS1 INCORRECT
5096 026730 023737 003510 003450 16$:  CMP      E.CS2,T.CS2  ;CHECK CS2 CORRECT
5097 026736 001401          BEQ      17$        ;YES, CHECK DATA
5098 026740 104152          ERROR   152        ;CS2 INCORRECT
5099 026742 005037 003624          CLR      WRDCNT     ;INITIALIZE WORD COUNT
5100 026746 012703 064412      MOV      #HEAD1,R3   ;GET ADDRESS OF DATA
5101 026752 012337 003522          MOV      (R3)+,E.DB  ;GET EXPECTED DATA
5102 026756 016237 000024 003462      MOV      RKDB(R2),T.DB ;GET ACTUAL DATA
5103 026764 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG. 1
5104 026772 016237 000010 003450      MOV      RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG. 2
5105 027000 022737 000002 003624      CMP      #2,WRDCNT   ;CHECK IF LAST WORD IN DATA BUFFER
5106 027006 001003          BNE      21$        ;NO, CHECK CS1
5107 027010 012737 000100 003510      MOV      #IR,E.CS2   ;STORE EXPECTED CS2
5108 027016 023737 003500 003440 21$:  CMP      E.CS1,T.CS1  ;CHECK CS1 CORRECT
5109 027024 001402          BEQ      22$        ;YES, CHECK CS2
5110 027026 104153          ERROR   153        ;CS1 INCORRECT
5111 027030 000421          BR      TST33      ;;GO ON TO NEXT TEST
5112
5113 027032 023737 003510 003450 22$:  CMP      E.CS2,T.CS2  ;CHECK CS2 CORRECT
5114 027040 001402          BEQ      23$        ;YES, CHECK DATA
5115 027042 104154          ERROR   154        ;CS2 INCORRECT
5116 027044 000413          BR      TST33      ;;GO ON TO NEXT TEST
5117
5118 027046 023737 003522 003462 23$:  CMP      E.DB,T.DB   ;CHECK IF DATA CORRECT
5119 027054 001401          BEQ      24$        ;YES, GET NEXT HEADER WORD
5120 027056 104155          ERROR   155        ;DATA INCORRECT
5121 027060 005237 003624          INC      WRDCNT     ;INCREMENT WORD COUNT
5122 027064 022737 000003 003624      CMP      #3,WRDCNT   ;CHECK IF ALL THREE WORDS CHECK
5123 027072 001327          BNE      20$        ;NO, GET NEXT WORD

```

.SBTTL \*\*MFM WRITE LOOPBACK TESTS

```

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

```

```
5137  
5138  
5139 027074 000004  
5140 027076 012737 000144 001200  
5141 027104 013702 001270  
5142 027110 012762 100000 000000  
5143 027116 012762 000040 000026  
5144 027124 012762 000140 000026  
5145 027132 012762 000040 000026  
5146 027140 012762 064714 000004  
5147 027146 012762 177777 000000  
5148 027154 012762 000027 000000  
5149 027162 012700 002364  
5150  
5151 027166 012762 000440 000026 1$:  
5152 027174 012762 000040 000026  
5153 027202 005300  
5154 027204 001370  
5155 027206 012700 000004  
5156 027212 012762 000240 000026  
5157 027220 012762 000640 000026 2$:  
5158 027226 012762 000240 000026  
5159 027234 005300  
5160 027236 001370  
5161 027240 012762 000040 000026  
5162 027246 012737 062040 003524  
5163  
5164 027254 012700 000002  
5165 027260 012762 000440 000026 3$:  
5166 027266 012762 000040 000026  
5167 027274 005300  
5168 027276 001370  
5169 027300 005037 003626  
5170 027304 005037 003612  
5171 027310 005037 003614  
5172 027314 005037 003616  
5173 027320 005037 003620  
5174 027324 012700 000764  
5175 027330 012737 06023 003170  
5176 027336 005037 003622  
5177 027342 004737 042570 5$:  
5178 027346 104170  
5179 027350 005237 003622  
5180 027354 005300  
5181 027356 001371  
5182 027360 042737 040000 003524  
5183 027366 052737 000100 003524  
5184 027374 012762 000140 000026  
5185 027402 016237 000026 003464  
5186 027410 023737 003524 003464  
5187 027416 001401  
5188 027420 104171  
5189 027422 042737 000100 003524 10$:  
5190 027430 052737 040000 003524  
5191 027436 012762 000040 000026  
5192 027444 016237 000026 003464
```

\*\*\*\*\*  
TST33: SCOPE  
MOV #100.,\$TIMES ;:DO 100. ITERATIONS  
MOV \$BASE,R2 ;LOAD RK611 BASE  
MOV #CLR,RKCS1(R2) ;CLEAR RK611  
MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE  
MOV #DMD!MSP,RKMR1(R2) ;INITIALIZE ROM  
MOV #DMD,RKMR1(R2)  
MOV #WRBUFF,RKBA(R2) ;ISSUE WRITE HEADER  
MOV #-1,RKWC(R2)  
MOV #WRHEAD,RKCS1(R2)  
MOV #<256.+48.+64.+256.+10.>\*2,R0 ;ISSUE ENOUGH CLOCKS  
; UNTIL READY FOR INDEX PULSE  
MOV #DMD!MCLK,RKMR1(R2) 1\$:  
MOV #DMD,RKMR1(R2)  
DEC R0  
BNE 1\$  
MOV #4.,R0 ;SIMULATE INDEX PULSE  
MOV #MIND!DMD,RKMR1(R2)  
MOV #DMD.MIND!MCLK,RKMR1(R2) 2\$:  
MOV #DMD!MIND,RKMR1(R2)  
DEC R0  
BNF 2\$  
MOV #DMD,RKMR1(R2)  
MOV #DMD!MEWD.ECCW!WRTGA,E.MR1 ;INITIALIZE EXPECTED  
; MAINT. REG  
MOV #2.,R0 ;WAIT FOR WRITE GATE  
MOV #DMD!MCLK,RKMR1(R2) 3\$:  
MOV #DMD,RKMR1(R2)  
DEC R0  
BNE 3\$  
CLR SECCNT ;CLEAR SECTOR COUNT  
CLR P1.BIT ;INITIALIZE BIT GENERATION  
CLR PR.BIT  
CLR M1.BIT  
CLR M2.BIT  
MOV #500.,R0 ;LOAD COUNT FOR 500 BITS  
MOV #EM230,EMW ;LOAD ERROR MESSAGE  
CLR BITCNT ;CLEAR BIT COUNT  
JSR PC,WRTBIT ;WRITE ONE BIT  
ERROR 170 ;ERROR IN WRITE  
INC BITCNT ;INCREMENT NUMBER OF BITS WRITTEN  
DEC R0 ;CHECK IF FINISHED  
BNE 5\$ ;NO, CONTINUE  
BIC #WRTGAT,E.MR1 ;GENERATE EXPECTED MR1  
BIS #MSP,E.MR1  
MOV #DMD!MSP,RKMR1(R2) ;RAISE SECTOR PULSE  
MOV RKMR1(R2),T.MR1 ;STORE MAINT. REG. 1  
MOV E.MR1,T.MR1 ;STORE MAINT REG 1  
BEQ 10\$ ;YES, LOWER SECTOR PULSE  
ERROR 171 ;WRITE GATE DID NOT RESET  
BIC #MSP,E.MR1 ;GENERATE EXPECTED MR1  
BIS #WRTGAT,E.MR1  
MOV #DMD,RKMR1(R2) ;RESET SECTOR PULSE  
MOV RKMR1(R2),T.MR1 ;STORE MAINT REG 1

5193 027452 023737 003524 003464  
 5194 027460 001401  
 5195 027462 104172  
 5196  
 5197  
 5198  
 5199  
 5200  
 5201  
 5202  
 5203  
 5204  
 5205  
 5206  
 5207  
 5208  
 5209  
 5210  
 5211  
 5212  
 5213  
 5214  
 5215 027464 000004  
 5216 027466 012737 000144 001200  
 5217 027474 013702 001270  
 5218 027500 012762 100000 000000  
 5219 027506 012762 000040 000026  
 5220 027514 012762 064412 000004  
 5221 027522 012762 177775 000002  
 5222 027530 012762 000027 000000  
 5223 027536 012700 000312  
 5224  
 5225 027542 012762 000440 000026 1\$:  
 5226 027550 012762 000040 000026  
 5227 027556 005300  
 5228 027560 001370  
 5229 027562 012700 000004  
 5230 027566 012762 000240 000026  
 5231 027574 012762 000640 000026 2\$:  
 5232 027602 012762 000240 000026  
 5233 027610 005300  
 5234 027612 001370  
 5235 027614 012762 000040 000026  
 5236 027622 012700 000010  
 5237 027626 012762 000440 000026 3\$:  
 5238 027634 012762 000040 000026  
 5239 027642 005300  
 5240 027644 001370  
 5241 027646 012762 000140 000026  
 5242 027654 012762 000040 000026  
 5243 027662 005037 003626  
 5244 027666 012737 060475 003170  
 5245 027674 012737 062040 003524  
 5246  
 5247 027702 005037 003612  
 5248 027706 005037 003614

```

CMP      E.MR1,T.MR1      ;CHECK MR1 CORRECT
BEQ      TST34            ;:YES, GO ON TO NEXT TEST
ERROR    172              ;WRITE GATE DID NOT SET

*****
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:

      177777
      000000
      177777

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

*****
TST34:  SLOPE
      MOV      #100, $TIMES      ;;DO 100. ITERATIONS
      MOV      $BASE,R2          ;:LOAD RK611 BASE
      MOV      #CLR,RKCS1(R2)    ;:CLEAR RK611
      MOV      #DMD,RKMR1(R2)    ;:PUT RK611 IN DIAGNOSTIC MODE
      MOV      #-3,RKWC(R2)      ;:ISSUE WRITE HEADER
      MOV      #WRHEAD,RKCS1(R2)
      MOV      #50.*4+2,R0       ;:ISSUE ENOUGH CLOCKS UNTIL
      ; READY FOR INDEX PULSE
1$:     MOV      #DMD!MCLK,RKMR1(R2)
      MOV      #DMD,RKMR1(R2)
      DEC      R0
      BNE     1$
      MOV      #4,R0             ;:ISSUE INDEX PULSE
2$:     MOV      #DMD!MIND,RKMR1(R2)
      MOV      #DMD!MIND!MCLK,RKMR1(R2)
      DEC      R0
      BNE     2$
3$:     MOV      #DMD,RKMR1(R2)
      MOV      #8,R0             ;:WAIT FOR WRITE GATE
      MOV      #DMD!MCLK,RKMR1(R2)
      MOV      #DMD,RKMR1(R2)
      DEC      R0
      BNE     3$
      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!WRITGAT,E.MR1 ;:INITIALIZE EXPECTED
      ; MAINT REG 1
      CLR      P1.BIT            ;:INITIALIZE BIT GENERATION
      CLR      PR.BIT
  
```

```

5249 027712 005037 003616 CLR M1.BIT
5250 027716 005037 003620 CLR M2.BIT
5251 027722 012700 000400 MOV #256.,RC ;SIMULATE SYNCH
5252 027726 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5253 027732 004737 042570 5$: JSR PC,WRTBIT ;WRITE ONE BIT
5254 027736 104170 ERROR 170 ;DATA INCORRECT
5255 027740 005237 003622 INC BITCNT
5256 027744 005300 DEC R0 ;CHECK IF READY FOR DATA
5257 027746 001371 BNE 5$ ;NO, GENERATE NEXT BIT
5258 027750 012737 000001 003612 MOV #1,P1.BIT ;PUT IN SYNCH BIT
5259 027756 004737 042570 JSR PC,WRTBIT
5260 027762 104170 ERROR 170 ;DATA INCORRECT
5261 027764 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
5262 027770 012737 060541 003170 MOV #EM234,EMW ;LOAD ERROR MESSAGE
5263 027776 012703 064412 MOV #HEAD1,R3 ;LOAD ADDRESS OF DATA
5264 030002 012700 000003 MOV #3,R0 ;LOAD NUMBER WORDS IN HEADER
5265 030006 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
5266 030010 012701 000020 MOV #16.,R1 ;LOAD BIT COUNT
5267 030014 013737 003616 003620 12$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5268 030022 013737 003614 003616 MOV PR.BIT,M1.BIT
5269 030030 013737 003612 003614 MOV P1.BIT,PR.BIT
5270 030036 006004 ROR R4 ;SHIFT IN NEXT BIT
5271 030040 103403 BCS 14$ ;CHECK IF ONE
5272 030042 005037 003612 CLR P1.BIT ;ZERO
5273 030046 000403 BR 15$ ;CLOCK IN BIT
5274
5275 030050 012737 000001 003612 14$: MOV #1,P1.BIT ;ONE
5276 030056 004737 042570 15$: JSR PC,WRTBIT ;WRITE BIT
5277 030062 104170 ERROR 170 ;BIT INCORRECT
5278 030064 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
5279 030070 005301 DEC R1 ;CHECK IF WORD FINISHED
5280 030072 001350 BNE 12$ ;NO, CONTINUE WITH NEXT BIT
5281 030074 005300 DEC R0 ;CHECK IF HEADER COMPLETE
5282 030076 001343 BNE 10$ ;NO, GET NEXT WORD
5283 030100 012701 000020 MOV #16.,R1 ;LOAD BIT COUNT FOR NEXT WORD
5284 030104 013737 003616 003620 18$: MOV M1.BIT,M2.BIT ;SHIFT BITS
5285 030112 013737 003614 003616 MOV PR.BIT,M1.BIT
5286 030120 013737 003612 003614 MOV P1.BIT,PR.BIT
5287 030126 005037 003612 CLR P1.BIT
5288 030132 004737 042570 JSR PC,WRTBIT ;WRITE ZERO
5289 030136 104170 ERROR 170 ;BIT INCORRECT
5290 030140 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
5291 030144 005301 DEC R1 ;CHECK IF FINISHED
5292 030146 001356 BNE 18$ ;NO, CLOCK NEXT BIT
5293 030150 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5294 030156 012700 000004 MOV #4,R0
5295 030162 012762 000640 000026 20$: MOV #DMD!MIND!MCLK,RKMR1(R2)
5296 030170 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5297 030176 005300 DEC R0
5298 030200 001370 BNE 20$
5299 030202 012762 000040 000026 MOV #DMD,RKMR1(R2)
5300 030210 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
5301 030216 012737 022040 003524 MOV #MEWD!ECCW.DMD,E.MR1 ;LOAD EXPECTED MR1
5302 030224 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5303 030232 001401 BEQ 25$ ;YES, CHECK IF READY SET
5304 030234 104173 ERROR 173 ;MAINT REG 1 INCORRECT

```

```

5305 030236 012700 000010      25$: MOV #8,R0 ;FINISH COMMAND
5306 030242 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
5307 030250 012762 000040 000026 MOV #DMD,RKMR1(R2)
5308 030256 005300 DEC R0
5309 030260 001370 BNE 26$
5310 030262 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5311 030270 012737 000226 003500 MOV #RDY.WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5312 030276 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5313 030304 001401 BEQ TST35 ;:YES, GO ON TO NEXT TEST
5314 030306 104174 ERROR 174 ;CS1 INCORRECT
5315
5316
5317
5318
5319
5320
5321
5322
5323
5324
5325
5326
5327
5328
5329
5330
5331
5332
5333
5334

```

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

\*\*\*\*\*  
 \*TST35: SCOPE  
 \*\*\*\*\*

```

5335 030310 000004      TST35: MOV #100,$TIMES ;:DO 100. ITERATIONS
5336 030312 012737 000144 001200 MOV $BASE,R2 ;:LOAD RK611 BASE
5337 030320 013702 001270 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
5338 030324 012762 100000 000000 MOV #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
5339 030332 012762 000040 000026 MOV #HEAD2,RKBA(R2) ;:ISSUE WRITE HEADER
5340 030340 012762 064420 000004 MOV #-3,F wC(R2)
5341 030346 012762 177775 000002 MOV #WRHEAD,RKCS1(R2)
5342 030354 012762 000027 000000 MOV #50.*4+2,R0 ;:ISSUE ENOUGH CLOCKS UNTIL
5343 030362 012700 000312 ;: READY FOR INDEX PULSE
5344
5345 030366 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
5346 030374 012762 000040 000026 MOV #DMD,RKMR1(R2)
5347 030402 005300 DEC R0
5348 030404 001370 BNE 1$
5349 030406 012700 000004 MOV #4,R0 ;:ISSUE INDEX PULSE
5350 030412 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5351 030420 012762 000640 000026 2$: MOV #DMD!MIND!MCLK,RKMR1(R2)
5352 030426 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
5353 030434 005300 DEC R0
5354 030436 001370 BNE 2$
5355 030440 012762 000040 000026 MOV #DMD,RKMR1(R2)
5356 030446 012700 000010 MOV #8,R0 ;:WAIT FOR WRITE GATE
5357 030452 012762 000440 000026 3$: MOV #DMD!MCLK,RKMR1(R2)
5358 030460 012762 000040 000026 MOV #DMD,RKMR1(R2)
5359 030466 005300 DEC R0
5360 030470 001370 BNE 3$

```

5361	030472	012762	000140	000026		MOV	#DMD!MSP,RKMR1(R2)	;SIMULATE SECTOR PULSE
5362	030500	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
5363	030506	005037	003626			CLR	SECCNT	;INITIALIZE SECTOR COUNT
5364	030512	012737	060475	003170		MOV	#EM233,EMW	;LOAD ERROR MESSAGE
5365	030520	012737	062040	003524		MOV	#DMD!MEWD.ECCW!WRTGAT,E.MR1	;INITIALIZE EXPECTED
5366								; MAINT REG 1
5367	030526	005037	003612			CLR	P1.BIT	;INITIALIZE BIT GENERATION
5368	030532	005037	003614			CLR	PR.BIT	
5369	030536	005037	003616			CLR	M1.BIT	
5370	030542	005037	003620			CLR	M2.BIT	
5371	030546	012700	000400			MOV	#256.,R0	;SIMULATE SYNCH
5372	030552	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
5373	030556	004737	042570		5\$:	JSR	PC,WRTBIT	;WRITE ONE BIT
5374	030562	104170				ERROR	170	;DATA INCORRECT
5375	030564	005237	003622			INC	BITCNT	
5376	030570	005300				DEC	R0	;CHECK IF READY FOR DATA
5377	030572	001371				BNE	5\$	;NO, GENERATE NEXT BIT
5378	030574	012737	000001	003612		MOV	#1,P1.BIT	;PUT IN SYNCH BIT
5379	030602	004737	042570			JSR	PC,WRTBIT	
5380	030606	104170				ERROR	170	;DATA INCORRECT
5381	030610	005037	003622			CLR	BITCNT	;INITIALIZE BIT COUNT
5382	030614	012737	060541	003170		MOV	#EM234,EMW	;LOAD ERROR MESSAGE
5383	030622	012703	064420			MOV	#HEAD2,R3	;LOAD ADDRESS OF DATA
5384	030626	012700	000003			MOV	#3,R0	;LOAD NUMBER WORDS IN HEADER
5385	030632	012304			10\$:	MOV	(R3)+,R4	;GET NEXT WORD
5386	030634	012701	000020			MOV	#16.,R1	;LOAD BIT COUNT
5387	030640	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
5388	030646	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5389	030654	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5390	030662	006004				ROR	R4	;SHIFT IN NEXT BIT
5391	030664	103403				BCS	14\$	;CHECK IF ONE
5392	030666	005037	003612			CLR	P1.BIT	;ZERO
5393	030672	000403				BR	15\$	;CLOCK IN BIT
5394								
5395	030674	012737	000001	003612	14\$:	MOV	#1,P1.BIT	;ONE
5396	030702	004737	042570		15\$:	JSR	PC,WRTBIT	;WRITE BIT
5397	030706	104170				ERROR	170	;BIT INCORRECT
5398	030710	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT
5399	030714	005301				DEC	R1	;CHECK IF WORD FINISHED
5400	030716	001350				BNE	12\$	;NO, CONTINUE WITH NEXT BIT
5401	030720	005300				DEC	R0	;CHECK IF HEADER COMPLETE
5402	030722	001343				BNE	10\$	;NO, GET NEXT WORD
5403	030724	012701	000020			MOV	#16.,R1	;LOAD BIT COUNT FOR NEXT WORD
5404	030730	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT	;SHIFT BITS
5405	030736	013737	003614	003616		MOV	PR.BIT,M1.BIT	
5406	030744	013737	003612	003614		MOV	P1.BIT,PR.BIT	
5407	030752	005037	003612			CLR	P1.BIT	
5408	030756	004737	042570			JSR	PC,WRTBIT	;WRITE ZERO
5409	030762	104170				ERROR	170	;BIT INCORRECT
5410	030764	005237	003622			INC	BITCNT	;INCREMENT BIT COUNT
5411	030770	005301				DEC	R1	;CHECK IF FINISHED
5412	030772	001356				BNE	18\$	;NO, CLOCK NEXT BIT
5413	030774	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	;SIMULATE INDEX
5414	031002	012700	000004			MOV	#4,R0	
5415	031006	012762	000640	000026	20\$:	MOV	#DMD!MIND.MCLK,RKMR1(R2)	
5416	031014	012762	000240	000026		MOV	#DMD!MIND,RKMR1(R2)	



```

5417 031022 005300      DEC      R0
5418 031024 001370      BNE      20$
5419 031026 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5420 031034 016237 000026 003464      MOV      RKMR1(R2),T,MR1 ;GET MAINT REG 1
5421 031042 012737 022040 003524      MOV      #MEWD!ECCW!DMD,E,MR1 ;LOAD EXPECTED MR1
5422 031050 023737 003524 003464      CMP      E,MR1,T,MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5423 031056 001401      BEQ      25$ ;YES, CHECK IF READY SET
5424 031060 104173      ERROR   173 ;MAINT REG 1 INCORREC
5425 031062 012700 000010      MOV      #8,R0 ;FINISH COMMAND
5426 031066 012762 000440 000026 25$:      MOV      #DMD!MCLK,RKMR1(R2)
5427 031074 012762 000040 000026 26$:      MOV      #DMD,RKMR1(R2)
5428 031102 005300      DEC      R0
5429 031104 001370      BNE      26$
5430 031106 016237 000000 003440      MOV      RKCS1(R2),T,CS1 ;GET COMMAND AND STATUS REG 1
5431 031114 012737 000226 003500      MOV      #RDY.WRHEAD&<^C<GO>>,E,CS1 ;LOAD EXPECTED CS1
5432 031122 023737 003500 003440      CMP      E,CS1,T,CS1 ;CHECK IF CS1 CORRECT
5433 031130 001401      BEQ      TST36 ;:YES, GO ON TO NEXT TEST
5434 031132 104174      ERROR   174 ;CS1 INCORRECT
5435
5436
5437
5438
5439
5440
5441
5442
5443
5444
5445
5446
5447
5448
5449
5450
5451
5452
5453

```

\*\*\*\*\*

\*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 PULSE.  
 \* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

\*\*\*\*\*

```

5454 031134 000004      TST36:  SCOPE
5455 031136 012737 000440 001200      MOV      #100,$TIMES ;:DO 100. ITERATIONS
5456 031144 013702 001270      MOV      $BASE,R2 ;:LOAD RK611 BASE
5457 031150 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;:CLEAR RK611
5458 031156 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
5459 031164 012762 064426 000004      MOV      #HEAD3,RKBA(R2) ;:ISSUE WRITE HEADER
5460 031172 012762 177775 000002      MOV      #-3,RKWC(R2)
5461 031200 012762 000027 000000      MOV      #WRHEAD,RKCS1(R2)
5462 031206 012700 000312      MOV      #50.*4+2,R0 ;:ISSUE ENOUGH CLOCKS UNTIL
5463 ;: READY FOR INDEX PULSE
5464 031212 012762 000440 000026 1$:      MOV      #DMD.MCLK,RKMR1(R2)
5465 031220 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5466 031226 005300      DEC      R0
5467 031230 001370      BNE      1$
5468 031232 012700 000004      MOV      #4,R0 ;:ISSUE INDEX PULSE
5469 031236 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
5470 031244 012762 000640 000026 2$:      MOV      #DMD:MIND!MCLK,RKMR1(R2)
5471 031252 012762 000240 000026      MOV      #DMD.MIND,RKMR1(R2)
5472 031260 005300      DEC      R0

```

```

5473 031262 001370      BNE      2$
5474 031264 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5475 031272 012700 000010      MOV      #8.,R0          ;WAIT FOR WRITE GATE
5476 031276 012762 000440 000026 3$:      MOV      #DMD!MCLK,RKMR1(R2)
5477 031304 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5478 031312 005300      DEC      R0
5479 031314 001370      BNE      3$
5480 031316 012762 000140 000026      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SFCTOR PULSE
5481 031324 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5482 031332 005037 003626      CLR      SECCNT          ;INITIALIZE SECTOR COUNT
5483 031336 012737 060475 003170      MOV      #EM233,EMW      ;LOAD ERROR MESSAGE
5484 031344 012737 062040 003524      MOV      #DMD!MEWD!ECCW'WRTGAT,E.MR1 ;INITIALIZE EXPECTED
5485                                     ; MAINT REG 1
5486 031352 005037 003612      CLR      P1.BIT          ;INITIALIZE BIT GENERATION
5487 031356 005037 003614      CLR      PR.BIT
5488 031362 005037 003616      CLR      M1.BIT
5489 031366 005037 003620      CLR      M2.BIT
5490 031372 012700 000400      MOV      #256.,R0        ;SIMULATE SYNCH
5491 031376 005037 003622      CLR      BITCNT          ;INITIALIZE BIT COUNT
5492 031402 004737 042570      JSR      PC,WRTBIT       ;WRITE ONE BIT
5493 031406 104170      ERROR    170             ;DATA INCORRECT
5494 031410 005237 003622      INC      BITCNT
5495 031414 005300      DEC      R0              ;CHECK IF READY FOR DATA
5496 031416 001371      BNE      5$              ;NO, GENERATE NEXT BIT
5497 031420 012737 000001 003612      MOV      #1,P1.BIT       ;PUT IN SYNCH BIT
5498 031426 004737 042570      JSR      PC,WRTBIT
5499 031432 104170      ERROR    170             ;DATA INCORRECT
5500 031434 005037 003622      CLR      BITCNT          ;INITIALIZE BIT COUNT
5501 031440 012737 060541 003170      MOV      #EM234,EMW      ;LOAD ERROR MESSAGE
5502 031446 012703 064426      MOV      #HEAD3,R3       ;LOAD ADDRESS OF DATA
5503 031452 012700 000003      MOV      #3,R0           ;LOAD NUMBER WORDS IN HEADER
5504 031456 012304      MOV      (R3)+,R4        ;GET NEXT WORD
5505 031460 012701 000020      MOV      #16.,R1        ;LOAD BIT COUNT
5506 031464 013737 003616 003620 12$:      MOV      M1.BIT,M2.BIT   ;SHIFT BITS
5507 031472 013737 003614 003616      MOV      PR.BIT,M1.BIT
5508 031500 013737 003612 003614      MOV      P1.BIT,PR.BIT
5509 031506 006004      ROR      R4              ;SHIFT IN NEXT BIT
5510 031510 103403      BCS      14$             ;CHECK IF ONE
5511 031512 005037 003612      CLR      P1.BIT         ;ZERO
5512 031516 000403      BR       15$            ;CLOCK IN BIT
5513
5514 031520 012737 000001 003612 14$:      MOV      #1,P1.BIT       ;ONE
5515 031526 004737 042570 15$:      JSR      PC,WRTBIT       ;WRITE BIT
5516 031532 104170      ERROR    170             ;BIT INCORRECT
5517 031534 005237 003622      INC      BITCNT          ;INCREMENT BIT COUNT
5518 031540 005301      DEC      R1              ;CHECK IF WORD FINISHED
5519 031542 001350      BNE      12$            ;NO, CONTINUE WITH NEXT BIT
5520 031544 005300      DEC      R0              ;CHECK IF HEADER COMPLETE
5521 031546 001343      BNE      10$            ;NO, GET NEXT WORD
5522 031550 012701 000020      MOV      #16.,R1        ;LOAD BIT COUNT FOR NEXT WORD
5523 031554 013737 003616 003620 18$:      MOV      M1.BIT,M2.BIT   ;SHIFT BITS
5524 031562 013737 003614 003616      MOV      PR.BIT,M1.BIT
5525 031570 013737 003612 003614      MOV      P1.BIT,PR.BIT
5526 031576 005037 003612      CLR      P1.BIT
5527 031602 004737 042570      JSR      PC,WRTBIT       ;WRITE ZERO
5528 031606 104170      ERROR    170             ;BIT INCORRECT

```

```

5529 031610 005237 003622      INC      BITCNT      ;INCREMENT BIT COUNT
5530 031614 005301              DEC      R1          ;CHECK IF FINISHED
5531 031616 001356              BNE     18$         ;NO, CLOCK NEXT BIT
5532 031620 012762 000240 000026  MOV     #DMD.MIND,RKMR1(R2) ;SIMULATE INDEX
5533 031626 012700 000004              MOV     #4,R0
5534 031632 012762 000640 000026 20$:  MOV     #DMD!MIND!MCLK,RKMR1(R2)
5535 031640 012762 000240 000026  MOV     #DMD!MIND,RKMR1(R2)
5536 031646 005300              DEC      R0
5537 031650 001370              BNE     20$
5538 031652 012762 000040 000026  MOV     #DMD,RKMR1(R2)
5539 031660 016237 000026 003464  MOV     RKMR1(R2),T.MR1 ;GET MAINT REG 1
5540 031666 012737 022040 003524  MOV     #MEWD!ECCW.DMD,E.MR1 ;LOAD EXPECTED MR1
5541 031674 023737 003524 003464  CMP     E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5542 031702 001401              BEQ     25$         ;YES, CHECK IF READY SET
5543 031704 104173              ERROR   173        ;MAINT REG 1 INCORRECT
5544 031706 012700 000010              MOV     #8.,R0 ;FINISH COMMAND
5545 031712 012762 000440 000026 25$:  MOV     #DMD.MCLK,RKMR1(R2)
5546 031720 012762 000040 000026 26$:  MOV     #DMD,RKMR1(R2)
5547 031726 005300              DEC      R0
5548 031730 001370              BNE     26$
5549 031732 016237 000000 003440  MOV     RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5550 031740 012737 000226 003500  MOV     #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5551 031746 023737 003500 003440  CMP     E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5552 031754 001401              BEQ     TST37      ;:YES, GO ON TO NEXT TEST
5553 031756 104174              ERROR   174        ;CS1 INCORRECT

```

```

5554
5555
5556
5557
5558
5559
5560
5561
5562
5563
5564
5565
5566
5567
5568
5569
5570
5571
5572

```

\*\*\*\*\*  
 \*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  
 \* 022222  
 \* 111111  
 \*  
 \* MAKE SURE THAT READY COMES UP AFTER THE SECOND INDEX MODE.  
 \* CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.  
 \*  
 \*\*\*\*\*

```

5573 031760 000004              TST37: SCOPE
5574 031762 012737 000144 001200  MOV     #100.,$TIMES ;:DO 100. ITERATIONS
5575 031770 013702 001270              MOV     $BASE,R2 ;LOAD RK611 BASE
5576 031774 012762 100000 000000  MOV     #CLR,RKCS1(R2) ;CLEAR RK611
5577 032002 012762 000040 000026  MOV     #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
5578 032010 012762 064442 000004  MOV     #HEAD4,RKBA(R2) ;ISSUE WRITE HEADER
5579 032016 012762 177775 000002  MOV     #-3,RKWC(R2)
5580 032024 012762 000027 000000  MOV     #WRHEAD,RKCS1(R2)
5581 032032 012700 000312              MOV     #50.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5582                                ; READY FOR INDEX PULSE
5583 032036 012762 000440 000026 1$:  MOV     #DMD!MCLK,RKMR1(R2)
5584 032044 012762 000040 000026  MOV     #DMD,RKMR1(R2)

```

```

5585 032052 005300      DEC      R0
5586 032054 001370      BNE      1$
5587 032056 012700 000004  MOV      #4,R0      ;ISSUE INDEX PULSE
5588 032062 012762 000240 000026  MOV      #DMD.MIND,RKMR1(R2)
5589 032070 012762 000640 000026 2$: MOV      #DMD!MIND.MCLK,RKMR1(R2)
5590 032076 012762 000240 000026  MOV      #DMD!MIND,RKMR1(R2)
5591 032104 005300      DEC      R0
5592 032106 001370      BNE      2$
5593 032110 012762 000040 000026  MOV      #DMD,RKMR1(R2)
5594 032116 012700 000010  MOV      #8,R0      ;WAIT FOR WRITE GATE
5595 032122 012762 000440 000026 3$: MOV      #DMD.MCLK,RKMR1(R2)
5596 032130 012762 000040 000026  MOV      #DMD,RKMR1(R2)
5597 032136 005300      DEC      R0
5598 032140 001370      BNE      3$
5599 032142 012762 000140 000026  MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5600 032150 012762 000040 000026  MOV      #DMD,RKMR1(R2)
5601 032156 005037 003626  CLR      SECCNT      ;INITIALIZE SECTOR COUNT
5602 032162 012737 060475 003170  MOV      #EM233,EMW      ;LOAD ERROR MESSAGE
5603 032170 012737 062040 003524  MOV      #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
5604                                     ; MAINT REG 1
5605 032176 005037 003612  CLR      P1.BIT      ;INITIALIZE BIT GENERATION
5606 032202 005037 003614  CLR      PR.BIT
5607 032206 005037 003616  CLR      M1.BIT
5608 032212 005037 003620  CLR      M2.BIT
5609 032216 012700 000400  MOV      #256,R0      ;SIMULATE SYNCH
5610 032222 005037 003622  CLR      BITCNT      ;INITIALIZE BIT COUNT
5611 032226 004737 042570 5$: JSR      PC,WRTBIT      ;WRITE ONE BIT
5612 032232 104170  ERROR    170      ;DATA INCORRECT
5613 032234 005237 003622  INC      BITCNT
5614 032240 005300      DEC      R0      ;CHECK IF READY FOR DATA
5615 032242 001371      BNE      5$      ;NO, GENERATE NEXT BIT
5616 032244 012737 000001 003612  MOV      #1,P1.BIT      ;PUT IN SYNCH BIT
5617 032252 004737 042570  JSR      PC,WRTBIT
5618 032256 104170  ERROR    170      ;DATA INCORRECT
5619 032260 005037 003622  CLR      BITCNT      ;INITIALIZE BIT COUNT
5620 032264 012737 060541 003170  MOV      #EM234,EMW      ;LOAD ERROR MESSAGE
5621 032272 012703 064442  MOV      #HEAD4,R3      ;LOAD ADDRESS OF DATA
5622 032276 012700 000003  MOV      #3,R0      ;LOAD NUMBER WORDS IN HEADER
5623 032302 012304 10$: MOV      (R3)+,R4      ;GET NEXT WORD
5624 032304 012701 000020  MOV      #16,R1      ;LOAD BIT COUNT
5625 032310 013737 003616 003620 12$: MOV      M1.BIT,M2.BIT      ;SHIFT BITS
5626 032316 013737 003614 003616  MOV      PR.BIT,M1.BIT
5627 032324 013737 003612 003614  MOV      P1.BIT,PR.BIT
5628 032332 006004  ROR      R4      ;SHIFT IN NEXT BIT
5629 032334 103403  BCS      14$      ;CHECK IF ONE
5630 032336 005037 003612  CLR      P1.BIT      ;ZERO
5631 032342 000403  BR       15$      ;CLOCK IN BIT
5632
5633 032344 012737 000001 003612 14$: MOV      #1,P1.BIT      ;ONE
5634 032352 004737 042570 15$: JSR      PC,WRTBIT      ;WRITE BIT
5635 032356 104170  ERROR    170      ;BIT INCORRECT
5636 032360 005237 003622  INC      BITCNT      ;INCREMENT BIT COUNT
5637 032364 005301  DEC      R1      ;CHECK IF WORD FINISHED
5638 032366 001350  BNE      12$      ;NO, CONTINUE WITH NEXT BIT
5639 032370 005300  DEC      R0      ;CHECK IF HEADER COMPLETE
5640 032372 001343  BNE      10$      ;NO, GET NEXT WORD
  
```

```

5641 032374 012701 000020      MOV      #16.,R1      ;LOAD BIT COUNT FOR NEXT WORD
5642 032400 013737 003616 003620 18$:  MOV      M1.BIT,M2.BIT ;SHIFT BITS
5643 032406 013737 003614 003616      MOV      PR.BIT,M1.BIT
5644 032414 013737 003612 003614      MOV      P1.BIT,PR.BIT
5645 032422 005037 003612      CLR      P1.BIT
5646 032426 004737 042570      JSR      PC,WRTBIT    ;WRITE ZERO
5647 032432 104170      ERROR   170          ;BIT INCORRECT
5648 032434 005237 003622      INC      BITCNT      ;INCREMENT BIT COUNT
5649 032440 005301      DEC      R1          ;CHECK IF FINISHED
5650 032442 001356      BNE     18$         ;NO, CLOCK NEXT BIT
5651 032444 012762 000240 000026      MOV      #DMD.MIND,RKMR1(R2) ;SIMULATE INDEX
5652 032452 012700 000004      MOV      #4,R0
5653 032456 012762 000640 000026 20$:  MOV      #DMD.MIND!MCLK,RKMR1(R2)
5654 032464 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
5655 032472 005300      DEC      R0
5656 032474 001370      BNE     20$
5657 032476 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5658 032504 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;GET MAINT REG 1
5659 032512 012737 022040 003524      MOV      #MEWD!ECCW.DMD,E.MR1 ;LOAD EXPECTED MR1
5660 032520 023737 003524 003464      CMP      E.MR1,T.MR1  ;CHECK MR1 CORRECT (WRITE GATE RESET)
5661 032526 001401      BFQ     25$         ;YES, CHECK IF READY SET
5662 032530 104173      ERROR   173          ;MAINT REG 1 INCORRECT
5663 032532 012700 000010      MOV      #8.,R0      ;FINISH COMMAND
5664 032536 012762 000440 000026 26$:  MOV      #DMD!MCLK,RKMR1(R2)
5665 032544 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5666 032552 005300      DEC      R0
5667 032554 001370      BNE     26$
5668 032556 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5669 032564 012737 000226 003500      MOV      #RDY!WRHEAD&<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5670 032572 023737 003500 003440      CMP      E.CS1,T.CS1  ;CHECK IF CS1 CORRECT
5671 032600 001401      BEQ     TST40       ;:YES, GO ON TO NEXT TEST
5672 032602 104174      ERROR   174          ;CS1 INCORRECT

```

```

5673
5674
5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691

```

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

```

5692 032604 000004      TST40: SCOPE
5693 032606 012737 000144 001200      MOV      #100.,$TIMES ;:DO 100. ITERATIONS
5694 032614 013702 001270      MOV      $BASE,R2    ;LOAD RK611 BASE
5695 032620 012762 100000 000000      MOV      #CCLR,RKCS1(R2) ;CLEAR RK611
5696 032626 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE

```

```

5697 032634 012762 064450 000004      MOV      #HEAD5,RKBA(R2) ;ISSUE WRITE HEADER
5698 032642 012762 177775 000002      MOV      #-3,RKWC(R2)
5699 032650 012762 000027 000000      MOV      #WRHEAD,RKCS1(R2)
5700 032656 012700 000312      MOV      #50.*4+2,R0 ;ISSUE ENOUGH CLOCKS UNTIL
5701                                     ; READY FOR INDEX PULSE
5702 032662 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2)
5703 032670 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5704 032676 005300      DEC      R0
5705 032700 001370      RNE     1$
5706 032702 012700 000004      MOV      #4,R0 ;ISSUE INDEX PULSE
5707 032706 012762 000240 000026      MOV      #DMD.MIND,RKMR1(R2)
5708 032714 012762 000640 000026 2$:      MOV      #DMD.MIND.MCLK,RKMR1(R2)
5709 032722 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
5710 032730 005300      DEC      R0
5711 032732 001370      BNE     2$
5712 032734 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5713 032742 012700 000010      MOV      #8.,R0 ;WAIT FOR WRITE GATE
5714 032746 012762 000440 000026 3$:      MOV      #DMD!MCLK,RKMR1(R2)
5715 032754 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5716 032762 005300      DEC      R0
5717 032764 001370      BNE     3$
5718 032766 012762 000440 000026      MOV      #DMD.MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
5719 032774 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5720 033002 005037 003626      CLR     SECCNT ;INITIALIZE SECTOR COUNT
5721 033006 012737 060475 003170      MOV      #EM233,EMW ;LOAD ERROR MESSAGE
5722 033014 012737 062040 003524      MOV      #DMD.MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
5723                                     ; MAINT REG 1
5724 033022 005037 003612      CLR     P1.BIT ;INITIALIZE BIT GENERATION
5725 033026 005037 003614      CLR     PR.BIT
5726 033032 005037 003616      CLR     M1.BIT
5727 033036 005037 003620      CLR     M2.BIT
5728 033042 012700 000400      MOV      #256.,R0 ;SIMULATE SYNCH
5729 033046 005037 003622      CLR     BITCNT ;INITIALIZE BIT COUNT
5730 033052 004737 042570 5$:      JSR     PC,WRTBIT ;WRITE ONE BIT
5731 033056 104170      ERROR   170 ;DATA INCORRECT
5732 033060 005237 003622      INC     BITCNT
5733 033064 005300      DEC     R0 ;CHECK IF READY FOR DATA
5734 033066 001371      BNE     5$ ;NO, GENERATE NEXT BIT
5735 033070 012737 000001 003612      MOV      #1,P1.BIT ;PUT IN SYNCH BIT
5736 033076 004737 042570      JSR     PC,WRTBIT
5737 033102 104170      ERROR   170 ;DATA INCORRECT
5738 033104 005037 003622      CLR     BITCNT ;INITIALIZE BIT COUNT
5739 033110 012737 060541 003170      MOV      #EM234,EMW ;LOAD ERROR MESSAGE
5740 033116 012703 064450      MOV      #HEAD5,R3 ;LOAD ADDRESS OF DATA
5741 033122 012700 000003      MOV      #3,R0 ;LOAD NUMBER WORDS IN HEADER
5742 033126 012304 10$:      MOV      (R3)+,R4 ;GET NEXT WORD
5743 033130 012701 000020      MOV      #16.,R1 ;LOAD BIT COUNT
5744 033134 013737 003616 003620 12$:      MOV      M1.BIT,M2.BIT ;SHIFT BITS
5745 033142 013737 003614 003616      MOV      PR.BIT,M1.BIT
5746 033150 013737 003612 003614      MOV      P1.BIT,PR.BIT
5747 033156 006004      ROR     R4 ;SHIFT IN NEXT BIT
5748 033160 103403      BCS     14$ ;CHECK IF ONE
5749 033162 005037 003612      CLR     P1.BIT ;ZERO
5750 033166 000403      BR     15$ ;CLOCK IN BIT
5751
5752 033170 012737 000001 003612 14$:      MOV      #1,P1.BIT ;ONE
  
```

```

5753 033176 004737 042570      15$: JSR    PC,WRTBIT      ;WRITE BIT
5754 033202 104170                ERROR  170      ;BIT INCORRECT
5755 033204 005237 003622      INC    BITCNT        ;INCREMENT BIT COUNT
5756 033210 005301                DEC    R1            ;CHECK IF WORD FINISHED
5757 033212 001350                BNE   12$           ;NO, CONTINUE WITH NEXT BIT
5758 033214 005300                DEC    R0            ;CHECK IF HEADER COMPLETE
5759 033216 001343                BNE   10$           ;NO, GET NEXT WORD
5760 033220 012701 000020      MOV    #16,R1        ;LOAD BIT COUNT FOR NLXT WORD
5761 033224 013737 003616 003620 18$: MOV    M1.BIT,M2.BIT ;SHIFT BITS
5762 033232 013737 003614 003616 MOV    PR.BIT,M1.BIT
5763 033240 013737 003612 003614 MOV    P1.BIT,PR.BIT
5764 033246 005037 003612      CLR    P1.BIT
5765 033252 004737 042570      JSR    PC,WRTBIT      ;WRITE ZERO
5766 033256 104170                ERROR  170      ;BIT INCORRECT
5767 033260 005237 003622      INC    BITCNT        ;INCREMENT BIT COUNT
5768 033264 005301                DEC    R1            ;CHECK IF FINISHED
5769 033266 001356                BNE   18$           ;NO, CLOCK NEXT BIT
5770 033270 012762 000240 000026 MOV    #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5771 033276 012700 000004      MOV    #4,R0
5772 033302 012762 000640 000026 20$: MOV    #DMD!MIND!MCLK,RKMR1(R2)
5773 033310 012762 000240 000026 MOV    #DMD.MIND,RKMR1(R2)
5774 033316 005300                DEC    R0
5775 033320 001370                BNE   20$
5776 033322 012762 000040 000026 MOV    #DMD,RKMR1(R2)
5777 033330 016237 000026 003464 MOV    RKMR1(R2),T.MR1 ;GET MAINT REG 1
5778 033336 012737 022040 003524 MOV    #MEWD!ECCW.DMD,E.MR1 ;LOAD EXPECTED MR1
5779 033344 023737 003524 003464 CMP    E.MR1,T.MR1   ;CHECK MR1 CORRECT (WRITE GATE RESET)
5780 033352 001401                BEQ   25$           ;YES, CHECK IF READY SET
5781 033354 104173                ERROR  173      ;MAINT REG 1 INCORRECT
5782 033356 012700 000010 25$: MOV    #8,R0        ;FINISH COMMAND
5783 033362 012762 000440 000026 26$: MOV    #DMD.MCLK,RKMR1(R2)
5784 033370 012762 000040 000026 MOV    #DMD,RKMR1(R2)
5785 033376 005300                DEC    R0
5786 033400 001370                BNE   26$
5787 033402 016237 000000 003440 MOV    RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5788 033410 012737 000226 003500 MOV    #RDY!WRHEAD&<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5789 033416 023737 003500 003440 CMP    E.CS1,T.CS1   ;CHECK IF CS1 CORRECT
5790 033424 001401                BEQ   TST41        ;:YES, GO ON TO NEXT TEST
5791 033426 104174                ERROR  174      ;CS1 INCORRECT

```

```

5792
5793
5794 *****
5795 *TEST 41      WRITE LOOPBACK (PART 6)
5796
5797 *
5798 * CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
5799 * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
5800 * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
5801 * CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
5802 * INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
5803 * CONSISTING OF THE FOLLOWING DATA, AND INDEX PULSE:
5804 *
5805 *           155555
5806 *           066666
5807 *           155555
5808 *
5809 * MAKE SURE READY COMES UP AFTER SECOND INDEX PULSE.
5810 * CHECK FOR CORRECT WRITE ENCODED DATA AND PRECOMPENSATION.

```

```

5809
5810
5811 033430 000004
5812 033432 012737 000144 001200
5813 033440 013702 001270
5814 033444 012762 100000 000000
5815 033452 012762 000040 000026
5816 033460 012762 064456 000004
5817 033466 012762 177775 000002
5818 033474 012762 000027 000000
5819 033502 012700 000312
5820
5821 033506 012762 000440 000026 1$:
5822 033514 012762 000040 000026
5823 033522 005300
5824 033524 001370
5825 033526 012700 000004
5826 033532 012762 000240 000026
5827 033540 012762 000640 000026 2$:
5828 033546 012762 000240 000026
5829 033554 005300
5830 033556 001370
5831 033560 012762 000040 000026
5832 033566 012700 000010
5833 033572 012762 000440 000026 3$:
5834 033600 012762 000040 000026
5835 033606 005300
5836 033610 001370
5837 033612 012762 000140 000026
5838 033620 012762 000040 000026
5839 033626 005037 003626
5840 033632 012737 060475 003170
5841 033640 012737 062040 003524
5842
5843 033646 005037 003612
5844 033652 005037 003614
5845 033656 005037 003616
5846 033662 005037 003620
5847 033666 012700 000400
5848 033672 005037 003622
5849 033676 004737 042570 5$:
5850 033702 104170
5851 033704 005237 003622
5852 033710 005300
5853 033712 001371
5854 033714 012737 000001 003612
5855 033722 004737 042570
5856 033726 104170
5857 033730 005037 003622
5858 033734 012737 060541 003170
5859 033742 012703 064456
5860 033746 012700 000003
5861 033752 012304 10$:
5862 033754 012701 000020
5863 033760 013737 003616 003620 12$:
5864 033766 013737 003614 003616

```

```

:
:*****
:ST41: SCOPE
MOV #100.,$TIMES ;:DO 100. ITERATIONS
MOV $BASE,R2 ;:LOAD RK611 BASE
MOV #CLR,RKCS1(R2) ;:CLEAR RK611
MOV #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
MOV #HEAD6,RKBA(R2) ;:ISSUE WRITE HFADE
MOV #-3,RKWC(R2)
MOV #WRHEAD,RKCS1(R2)
MOV #50.*4+2,R0 ;:ISSUE ENOUGH CLOCKS UNTIL
;: READY FOR INDEX PULSE
MOV #DMD!MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV #4,R0 ;:ISSUE INDEX PULSE
MOV #DMD!MIND,RKMR1(R2)
MOV #DMD.MIND!MCLK,RKMR1(R2)
MOV #DMD!MIND,RKMR1(R2)
DFC R0
BNE 2$
MOV #DMD,RKMR1(R2)
MOV #8.,R0 ;:WAIT FOR WRITE GATE
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
CLR BITCNT ;:INITIALIZE BIT COUNT
JSR PC,WRTBIT ;:WRITE ONE BIT
ERROR 170 ;:DATA INCORRECT
INC BITCNT
DEC R0 ;:CHECK IF READY FOR DATA
BNE 5$ ;:NO, GENERATE NEXT BIT
MOV #1,P1.BIT ;:PUT IN SYNCH BIT
JSR PC,WRTBIT
ERROR 170 ;:DATA INCORRECT
CLR BITCNT ;:INITIALIZE BIT COUNT
MOV #EM234,EMW ;:LOAD ERROR MESSAGE
MOV #HEAD6,R3 ;:LOAD ADDRESS OF DATA
MOV #3,R0 ;:LOAD NUMBER WORDS IN HEADER
MOV (R3)+,R4 ;:GET NEXT WORD
MOV #16.,R1 ;:LOAD BIT COUNT
MOV M1.BIT,M2.BIT ;:SHIFT BITS
MOV PR.BIT,M1.BIT

```



```

5865 033774 013737 003612 003614      MOV      P1.BIT,PR.BIT
5866 034002 006004                ROR      R4          ;SHIFT IN NEXT BIT
5867 034004 103403                BCS      14$        ;CHECK IF ONE
5868 034006 005037 003612      CLR      P1.BIT     ;ZERO
5869 034012 000403                BR       15$        ;CLOCK IN BIT
5870
5871 034014 012737 000001 003612 14$:  MOV      #1,P1.BIT ;ONE
5872 034022 004737 042570 15$:  JSR      PC,WRTBIT ;WRITE BIT
5873 034026 104170                ERROR    170        ;BIT INCORRECT
5874 034030 005237 003622      INC      BITCNT     ;INCREMENT BIT COUNT
5875 034034 005301                DEC      R1          ;CHECK IF WORD FINISHED
5876 034036 001350                BNE     12$        ;NO, CONTINUE WITH NEXT BIT
5877 034040 005300                DEC      R0          ;CHECK IF HEADER COMPLETE
5878 034042 001343                BNE     10$        ;NO, GET NEXT WORD
5879 034044 012701 000020      MOV      #16,R1     ;LOAD BIT COUNT FOR NEXT WORD
5880 034050 013737 003616 003620 18$:  MOV      M1.BIT,M2.BIT ;SHIFT BITS
5881 034056 013737 003614 003616      MOV      PR.BIT,M1.BIT
5882 034064 013737 003612 003614      MOV      P1.BIT,PR.BIT
5883 034072 005037 003612      CLR      P1.BIT
5884 034076 004737 042570      JSR      PC,WRTBIT ;WRITE ZERO
5885 034102 104170                ERROR    170        ;BIT INCORRECT
5886 034104 005237 003622      INC      BITCNT     ;INCREMENT BIT COUNT
5887 034110 005301                DEC      R1          ;CHECK IF FINISHED
5888 034112 001356                BNF     18$        ;NO, CLOCK NEXT BIT
5889 034114 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
5890 034122 012700 000004      MOV      #4,R0
5891 034126 012762 000640 000026 20$:  MOV      #DMD.MIND!MCLK,RKMR1(R2)
5892 034134 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
5893 034142 005300                DEC      R0
5894 034144 001370                BNE     20$
5895 034146 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5896 034154 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;GET MAINT REG 1
5897 034162 012737 022040 003524      MOV      #MEWD.ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
5898 034170 023737 003524 003464      CMP      E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET)
5899 034176 001401                BEQ     25$        ;YES, CHECK IF READY SET
5900 034200 104173                ERROR    173        ;MAINT REG 1 INCORRECT
5901 034202 012700 000010 25$:  MOV      #8,R0      ;FINISH COMMAND
5902 034206 012762 000440 000026 26$:  MOV      #DMD!MCLK,RKMR1(R2)
5903 034214 012762 000040 000026      MOV      #DMD,RKMR1(R2)
5904 034222 005300                DEC      R0
5905 034224 001370                BNE     26$
5906 034226 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
5907 034234 012737 000226 003500      MOV      #RDY.WRHEAD&<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
5908 034242 023737 003500 003440      CMP      E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
5909 034250 001401                BEQ     TST42      ;YES, GO ON TO NEXT TEST
5910 034252 104174                ERROR    174        ;CS1 INCORRECT

```

```

5911
5912
5913 *****
5914 *TEST 42 WRITE LOOPBACK (PART 7)
5915 *
5916 * CLEAR THE RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
5917 * IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO AN RK06
5918 * IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, DRIVE 0.
5919 * CLOCK BOTH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE
5920 * INDEX PULSE, SECTOR PULSE, ONE THREE WORD HEADER
5921 * CONSISTING OF THE FOLLOWING DATA, AND INDEX PULSE:

```

```

5921
5922
5923
5924
5925
5926
5927
5928
5929
5930 034254 000004
5931 034256 012737 000144 001200
5932 034264 013702 001270
5933 034270 012762 100000 000000
5934 034276 012762 000040 000026
5935 034304 012762 064464 000004
5936 034312 012762 177775 000002
5937 034320 012762 000027 000000
5938 034326 012700 000312
5939
5940 034332 012762 000440 000026 1$:
5941 034340 012762 000040 000026
5942 034346 005300
5943 034350 001370
5944 034352 012700 000004
5945 034356 012762 000240 000026
5946 034364 012762 000640 000026 2$:
5947 034372 012762 000240 000026
5948 034400 005300
5949 034402 001370
5950 034404 012762 000040 000026
5951 034412 012700 000010
5952 034416 012762 000440 000026 3$:
5953 034424 012762 000040 000026
5954 034432 005300
5955 034434 001370
5956 034436 012762 000140 000026
5957 034444 012762 000040 000026
5958 034452 005037 003626
5959 034456 012737 060475 003170
5960 034464 012737 062040 003524
5961
5962 034472 005037 003612
5963 034476 005037 003614
5964 034502 005037 003616
5965 034506 005037 003620
5966 034512 012700 000400
5967 034516 005037 003622
5968 034522 004737 042570 5$:
5969 034526 104170
5970 034530 005237 003622
5971 034534 005300
5972 034536 001371
5973 034540 012737 000001 003612
5974 034546 004737 042570
5975 034552 104170
5976 034554 005037 003622

*****
TS42: SCOPE
MOV #100.,$TIMES ;:DO 100. ITERATIONS
MOV $BASE,R2 ;:LOAD RK611 BASE
MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
MOV #DMD,RKMR1(R2) ;:PUT RK611 IN DIAGNOSTIC MODE
MOV #HEAD7,RKBA(R2) ;:ISSUE WRITE HEADER
MOV #-3,RKWC(R2)
MOV #WRHEAD,RKCS1(R2)
MOV #50.*4+2,R0 ;:ISSUE ENOUGH CLOCKS UN'TIL
;: READY FOR INDEX PULSE
MOV #DMD.MCLK,RKMR1(R2)
MOV #DMD,RKMR1(R2)
DEC R0
BNE 1$
MOV #4,R0 ;:ISSUE INDEX PULSE
MOV #DMD.MIND,RKMR1(R2)
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
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,EMR1 ;:INITIALIZE EXPECTED
;: MAINT REG 1
CLR P1.BIT ;:INITIALIZE BIT GENERATION
CLR PR.BIT
CLR M1.BIT
CLR M2.BIT
MOV #256.,R0 ;:SIMULATE SYNCH
CLR BITCNT ;:INITIALIZE BIT COUNT
JSR PC,WRTBIT ;:WRITE ONE BIT
ERROR 170 ;:DATA INCORRECT.
INC BITCNT
DEC R0 ;:CHECK IF READY FOR DATA
BNE 5$ ;:NO, GENERATE NEXT BIT
MOV #1,P1.BIT ;:PUT IN SYNCH BIT
JSR PC,WRTBIT
ERROR 170 ;:DATA INCORRECT
CLR BITCNT ;:INITIALIZE BIT COUNT
  
```

```

5977 034560 012737 060541 003170      MOV      #EM234,EMW      ;LOAD ERPOR MESSAGE
5978 034566 012703 064464              MOV      #HEAD?,R3      ;LOAD ADDRESS OF DATA
5979 034572 012700 000003              MOV      #3,R0          ;LOAD NUMBER WORDS IN HEADER
5980 034576 012304              10$:    MOV      (R3)+,R4      ;GET NEXT WORD
5981 034600 012701 000020              MOV      #16.,R1       ;LOAD BIT COUNT
5982 034604 013737 003616 003620 12$:    MOV      M1.BIT,M2.BIT  ;SHIFT BITS
5983 034612 013737 003614 003616      MOV      PR.BIT,M1.BIT
5984 034620 013737 003612 003614      MOV      P1.BIT,PR.BIT
5985 034626 006004              ROR      R4             ;SHIFT IN NEXT BIT
5986 034630 103403              BCS     14$            ;CHECK IF ONE
5987 034632 005037 003612              CLR     P1.BIT        ;ZERO
5988 034636 000403              BR      15$            ;CLOCK IN BIT
5989
5990 034640 012737 000001 003612 14$:    MOV      #1,P1.BIT     ;ONE
5991 034646 004737 042570 15$:    JSR     PC,WRTBIT     ;WRITE BIT
5992 034652 104170              ERROR   170           ;BIT INCORRECT
5993 034654 005237 003622              INC     BITCNT        ;INCREMENT BIT COUNT
5994 034660 005301              DEC     R1             ;CHECK IF WORD FINISHED
5995 034662 001350              BNE     12$           ;NO, CONTINUE WITH NEXT BIT
5996 034664 005300              DEC     R0             ;CHECK IF HEADER COMPLETE
5997 034666 001343              BNE     10$           ;NO, GET NEXT WORD
5998 034670 012701 000020              MOV      #16.,R1       ;LOAD BIT COUNT FOR NEXT WORD
5999 034674 013737 003616 003620 18$:    MOV      M1.BIT,M2.BIT  ;SHIFT BITS
6000 034702 013737 003614 003616      MOV      PR.BIT,M1.BIT
6001 034710 013737 003612 003614      MOV      P1.BIT,PR.BIT
6002 034716 005037 003612              CLR     P1.BIT
6003 034722 004737 042570              JSR     PC,WRTBIT     ;WRITE ZERO
6004 034726 104170              ERROR   170           ;BIT INCORRECT
6005 034730 005237 003622              INC     BITCNT        ;INCREMENT BIT COUNT
6006 034734 005301              DEC     R1             ;CHECK IF FINISHED
6007 034736 001356              BNE     18$           ;NO, CLOCK NEXT BIT
6008 034740 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX
6009 034746 012700 000004              MOV      #4,R0
6010 034752 012762 000640 000026 20$:    MOV      #DMD!MIND!MCLK,RKMR1(R2)
6011 034760 012762 000240 000026      MOV      #DMD!MIND,RKMR1(R2)
6012 034766 005300              DEC     R0
6013 034770 001370              BNE     20$
6014 034772 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6015 035000 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;GET MAINT REG 1
6016 035006 012737 022040 003524      MOV      #MEWD.ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
6017 035014 023737 003524 003464      CMP     E.MR1,T.MR1   ;CHECK MR1 CORRECT (WRITE GATE RESET)
6018 035022 001401              BEQ     25$           ;YES, CHECK IF READY SET
6019 035024 104173              ERROR   173           ;MAINT REG 1 INCORRECT
6020 035026 012700 000010 25$:    MOV      #8.,R0       ;FINISH COMMAND
6021 035032 012762 000440 000026 26$:    MOV      #DMD!MCLK,RKMR1(R2)
6022 035040 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6023 035046 005300              DEC     R0
6024 035050 001370              BNE     26$
6025 035052 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6026 035060 012737 000226 003500      MOV      #RDY!WRHEAD&<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6027 035066 023737 003500 003440      CMP     E.CS1,T.CS1   ;CHECK IF CS1 CORRECT
6028 035074 001401              BEQ     TST43         ;:YES, GO ON TO NEXT TEST
6029 035076 104174              ERROR   174           ;CS1 INCORRECT

```

```

6030
6031
6032
:*****
: *TEST 43          WRITE TWO HEADERS

```

6033  
6034  
6035  
6036  
6037  
6038  
6039  
6040  
6041  
6042  
6043  
6044  
6045  
6046  
6047  
6048  
6049  
6050  
6051  
6052  
6053  
6054  
6055  
6056 035100 000004  
6057 035102 012737 000144 001200  
6058 035110 013702 001270  
6059 035114 012762 100000 000000  
6060 035122 012762 000040 000026  
6061 035130 012762 064412 000004  
6062 035136 012703 064412  
6063 035142 012762 177772 000002  
6064 035150 012762 000027 000000  
6065 035156 012700 000312  
6066  
6067 035162 012762 000440 000026 1\$:  
6068 035170 012762 000040 000026  
6069 035176 005300  
6070 035200 001370  
6071 035202 012700 000004  
6072 035206 012762 000240 000026  
6073 035214 012762 000640 000026 2\$:  
6074 035222 012762 000240 000026  
6075 035230 005300  
6076 035232 001370  
6077 035234 005037 003626  
6078 035240 012762 000040 000026  
6079 035246 012705 000002  
6080 035252 012700 000010  
6081 035256 012762 000440 000026 3\$:  
6082 035264 012762 000040 000026  
6083 035272 005300  
6084 035274 001370  
6085 035276 012762 000140 000026 4\$:  
6086 035304 012762 000040 000026  
6087 035312 012737 060475 003170  
6088 035320 012737 062040 003524

```

**
** CLEAR RK611 WITH A CONTROLLER CLEAR. PUT CONTROLLER
** IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO A. 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.
*****
TST43: SCOPE
MOV #100, $TIMES ; DO 100. ITERATIONS
MOV $BASE, R2 ; LOAD RK611 BASE
MOV #CLR, RKCS1(R2) ; CLEAR RK611
MOV #DMD, RKMR1(R2) ; PUT RK611 TO DIAGNOSTIC MODE
MOV #HEAD1, RKBA(R2) ; ISSUE WRITE HEADER
MOV #HEAD1, R3
MOV #-6, RKWC(R2)
MOV #WRHEAD, RKCS1(R2)
MOV #50.*4+2, R0 ; ISSUE ENOUGH CLOCKS UNTIL
; READY FOR INDEX PULSE
1$: 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$: MOV #DMD, MIND, MCLK, RKMR1(R2)
MOV #DMD, MIND, RKMP1(R2)
DEC R0
BNE 2$
CLR SECCNT ; CLEAR SECTOR COUNT
MOV #DMD, RKMR1(R2)
MOV #2, R5 ; LOAD NUMBER OF HEADERS
MOV #8, R0 ; WAIT FOR WRITE GATE
3$: MOV #DMD, MCLK, RKMR1(R2)
MOV #DMD, RKMR1(R2)
DEC R0
BNE 3$
4$: MOV #DMD, MSP, RKMR1(R2) ; SIMULATE SECTOR PULSE
MOV #DMD, RKMR1(R2)
MOV #EM233, EMW ; LOAD ERROR MESSAGE
MOV #DMD, MEWD, ECCW, WRTGAT, E.MR1 ; INITIALIZE EXPECTED

```

```
6089 ; MAINT REG 1
6090 035326 005037 003612 CLR P1.BIT
6091 035332 005037 003614 CLR PR.BIT
6092 035336 005037 003616 CLR M1.BIT
6093 035342 005037 003620 CLR M2.BIT
6094 035346 012700 000400 MOV #256.,R0 ;SIMULATE SYNCH
6095 035352 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
6096 035356 004737 042570 5$: JSR PC,WRTBIT ;WRITE ONE BIT
6097 035362 104170 ERROR 170 ;DATA INCORRECT
6098 035364 005237 003622 INC BITCNT
6099 035370 005300 DEC R0 ;CHECK IF READY FOR DATA
6100 035372 001371 BNE 5$ ;NO,GENERATE NEXT BIT
6101 035374 012737 000001 003612 MOV #1,P1.BIT ;PUT IN SYNCH BIT
6102 035402 004737 042570 JSR PC,WRTBIT
6103 035406 104170 ERROR 170 ;DATA INCORRECT
6104 035410 005037 003622 CLR BITCNT ;INITIALIZE BIT COUNT
6105 035414 012737 060541 003170 MOV #EM234,EMW ;LOAD ERROR MESSAGE
6106 035422 012700 000003 MOV #3,R0 ;LOAD NUMBER OF WORDS IN HEADER
6107 035426 012304 10$: MOV (R3)+,R4 ;GET NEXT WORD
6108 035430 012701 000020 MOV #16.,R1 ;LOAD BIT COUNT
6109 035434 013737 003616 003620 12$: MOV M1.BIT,M2.BIT ;SHIFT BITS
6110 035442 013737 003614 003616 MOV PR.BIT,M1.BIT
6111 035450 013737 003612 003614 MOV P1.BIT,PR.BIT
6112 035456 006004 ROR R4 ;SHIFT IN NEXT BIT
6113 035460 103403 BCS 14$ ;CHECK IF ONE
6114 035462 005037 003612 CLR P1.BIT ;ZERO
6115 035466 000403 BR 15$ ;CLOCK IN BIT
6116
6117 035470 012737 000001 003612 14$: MOV #1,P1.BIT ;ONE
6118 035476 004737 042570 5$: JSR PC,WRTBIT ;WRITE BIT
6119 035502 104170 ERROR 170 ;BIT INCORRECT
6120 035504 005237 003622 INC BITCNT ;INCREMENT BIT COUNT
6121 035510 005301 DEC R1 ;CHECK IF WORD FINISHED
6122 035512 001350 BNE 12$ ;NO, CONTINUE
6123 035514 005300 DEC R0 ;CHECK IF HEADER COMPLETE
6124 035516 001343 BNE 10$ ;NO, GET NEXT WORD
6125 035520 012701 000020 MOV #16.,R1 ;LOAD BIT COUNT FOR NEXT WORD
6126 035524 013737 003616 003620 18$: MOV M1.BIT,M2.BIT ;SHIFT BITS
6127 035532 013737 003614 003616 MOV PR.BIT,M1.BIT
6128 035540 013737 003612 003614 MOV P1.BIT,PR.BIT
6129 035546 005037 003612 CLR P1.BIT
6130 035552 004737 042570 JSR PC,WRTBIT ;WRITE ZERO
6131 035556 104170 ERROR 170 ;BIT INCORRECT
6132 035560 005237 003622 INC BITCNT ;INCREMENT
6133 035564 005301 DEC R1 ;CHECK IF READY FOR NEXT HEADER
6134 035566 001356 BNE 18$ ;NO, CONTINUE
6135 035570 005237 003626 INC SECCNT ;INCREMENT
6136 035574 005305 DEC R5 ;CHECK IF SECOND HEADER WRITTEN
6137 035576 001237 BNE 4$ ;NO, DO SECOND HEADER
6138 035600 012762 000240 000026 MOV #DMD.MIND,RKMR1(R2) ;SIMULATE INDEX PULSE
6139 035606 012700 000004 MOV #4,R0
6140 035612 012762 000640 000026 20$: MOV #DMD.MIND.MCLK,RKMR1(R2)
6141 035620 012762 000240 000026 MOV #DMD.MIND,RKMR1(R2)
6142 035626 005300 DEC R0
6143 035630 001370 BNE 20$
6144 035632 012762 000040 000026 MOV #DMD,RKMR1(R2)
```

```

6145 035640 016237 000026 003464 MOV RKMR1(R2),T.MR1 ;GET MAINT REG 1
6146 035646 012737 022040 003524 MOV #MEWD.ECCW!DMD,E.MR1 ;LOAD EXPECTED MR1
6147 035654 023737 003524 003464 CMP E.MR1,T.MR1 ;CHECK MR1 CORRECT (WRITE GATE RESET
6148 035662 001401 BEQ 25$ ;YES, CHECK IF READY SET
6149 035664 104173 ERROR 173 ;MAINT REG 1 INCORRECT
6150 035666 012700 000010 25$: MOV #8,R0 ;FINISH COMMAND
6151 035672 012762 000040 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
6152 035700 012762 000040 000026 MOV #DMD,RKMR1(R2)
6153 035706 005300 DEC R0
6154 035710 001370 BNE 26$
6155 035712 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6156 035720 012737 000226 003500 MOV #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6157 035726 023737 003500 003440 CMP E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
6158 035734 001401 BEQ TST44 ;:YES, GO ON TO NEXT TEST
6159 035736 104174 ERROR 174

```

```

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

```

```

6177 : MAKE SURE THE DATA SYNCH ANY OTHER BITS OF DATA AND
6178 : ECC FIELD ARE WRITTEN CORRECTLY.
6179 :
6180 :*****

```

```

6181 035740 000004 TST44: SCOPE
6182 035742 012737 000144 001200 MOV #100,$TIMES ;:DO 100. ITERATIONS
6183 035750 013702 001270 MOV $BASE,R2 ;LOAD RK611 BASE
6184 035754 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;CLEAR RK611
6185 035762 012762 000040 000026 MOV #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
6186 035770 012762 064426 000004 MOV #HEAD3,RKBA(R2) ;ISSUE READ HEADER
6187 035776 012703 064426 MOV #HEAD3,R3
6188 036002 012762 177772 000002 MOV #-2*3,RKWC(R2)
6189 036010 012762 000027 000000 MOV #WRHEAD,RKCS1(R2)
6190 036016 012700 000312 MOV #50.*4+2,R0 ;ISSUE CLOCKS UNTIL READY
6191 : FOR INDEX PULSE
6192 036022 012762 000440 000026 1$: MOV #DMD.MCLK,RKMR1(R2)
6193 036030 012762 000040 000026 MOV #DMD,RKMR1(R2)
6194 036036 005300 DEC R0
6195 036040 001370 BNE 1$
6196 036042 012700 000004 MOV #4,R0 ;ISSUE INDEX PULSE
6197 036046 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
6198 036054 012762 000640 000026 2$: MOV #DMD.MIND!MCLK,RKMR1(R2)
6199 036062 012762 000240 000026 MOV #DMD!MIND,RKMR1(R2)
6200 036070 005300 DEC R0

```

```

6201 036072 001370      BNE      2$
6202 036074 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6203 036102 012700 000010      MOV      #8,R0          ;WAIT FOR WRITE GATE
6204 036106 012762 000440 000026 3$:      MOV      #DMD.MCLK,RKMR1(R2)
6205 036114 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6206 036122 005300      DEC      R0
6207 036124 001370      BNE      3$
6208 036126 005037 003626      CLR      SECCNT        ;CLEAR SECTOR COUNT
6209 036132 012705 000002      MOV      #2,R5        ;LOAD NUMBER OF HEADERS
6210 036136 012762 000140 000026 4$:      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
6211 036144 012762 000040 000026      MOV      #DMD,RKMR1(R2)
6212 036152 012737 060475 003170      MOV      #EM233,EMW    ;LOAD ERROR MESSAGE
6213 036160 012737 062040 003524      MOV      #DMD!MEWD!ECCW!WRTGAT,E.MR1 ;INITIALIZE EXPECTED
6214                                     ; MAINT REG 1
6215 036166 005037 003612      CLR      P1.BIT
6216 036172 005037 003614      CLR      PR.BIT
6217 036176 005037 003616      CLR      M1.BIT
6218 036202 005037 003620      CLR      M2.BIT
6219
6220 036206 012700 000400      MOV      #256,R0      ;SIMULATE SYNCH
6221 036212 005037 003622      CLR      BITCNT      ;INITIALIZE BIT COUNT
6222 036216 004737 042570      JSR      PC,WRTBIT   ;WRITE ONE BIT
6223 036222 104170      ERROR    170        ;DATA INCORRECT
6224 036224 005237 003622      INC      BITCNT
6225 036230 005300      DEC      R0          ;CHECK IF READY FOR DATA
6226 036232 001371      BNE      5$          ;NO, GENERATE NEXT BIT
6227 036234 012737 000001 003612      MOV      #1,P1.BIT   ;PUT IN SYNCH BIT
6228 036242 004737 042570      JSR      PC,WRTBIT
6229 036246 104170      ERROR    170        ;DATA INCORRECT
6230 036250 005037 003622      CLR      BITCNT      ;INITIALIZE BIT COUNT
6231 036254 012737 060541 003170      MOV      #EM234,EMW  ;LOAD ERROR MESSAGE
6232 036262 012700 000003      MOV      #3,R0      ;LOAD NUMBER OF WORDS IN HEADER
6233 036266 012304      10$:      MOV      (R3)+,R4    ;GET NEXT WORD
6234 036270 012701 000020      MOV      #16,R1     ;LOAD BIT COUNT
6235 036274 013737 003616 003620 12$:      MOV      M1.BIT,M2.BIT ;SHIFT BITS
6236 036302 013737 003614 003616      MOV      PR.BIT,M1.BIT
6237 036310 013737 003612 003614      MOV      P1.BIT,PR.BIT
6238 036316 006004      ROR      R4          ;SHIFT IN NEXT BIT
6239 036320 103403      BCS      14$        ;CHECK IF ONE
6240 036322 005037 003612      CLR      P1.BIT     ;ZERO
6241 036326 000403      BR       15$        ;CLOCK IN BIT
6242
6243 036330 012737 000001 003612 14$:      MOV      #1,P1.BIT   ;ONE
6244 036336 004737 042570      15$:      JSR      PC,WRTBIT   ;WRITE BIT
6245 036342 104170      ERROR    170        ;BIT INCORRECT
6246 036344 005237 003622      INC      BITCNT      ;INCREMENT BIT COUNT
6247 036350 005301      DEC      R1          ;CHECK IF WORD FINISHED
6248 036352 001350      BNE      12$        ;NO, CONTINUE
6249 036354 005300      DEC      R0          ;CHECK IF HEADER COMPLETE
6250 036356 001343      BNE      10$        ;NO, GET NEXT WORD
6251 036360 012737 060773 003170      MOV      #EM237,EMW  ;LOAD ERROR MESSAGE
6252 036366 012701 000477      MOV      #64.+255,R1 ;LOAD COUNT
6253 036372 013737 003616 003620 18$:      MOV      M1.BIT,M2.BIT ;SHIFT BITS
6254 036400 013737 003614 003616      MOV      PR.BIT,M1.BIT
6255 036406 013737 003612 003614      MOV      P1.BIT,PR.BIT
6256 036414 005037 003612      CLR      P1.BIT
  
```

6257	036420	004737	042570			JSR	PC,WRTBIT	:WRITE ZERO
6258	036424	104170				ERROR	170	:BIT INCORRECT
6259	036426	005301				DEC	R1	:CHECK IF READY FOR DATA
6260	036430	001360				BNE	18\$	:NO, CONTINUE
6261	036432	012737	000001	003612		MOV	#1,P1.BIT	:SET SYNCH BIT
6262	036440	004737	042570			JSR	PC,WRTBIT	:WRITE SYNCH BIT
6263	036444	104170				ERROR	170	:SYNCH BIT INCORRECT
6264	036446	012737	061041	003170		MOV	#EM238,EMW	:LOAD ERROR MESSAGE
6265	036454	005037	003622			CLR	BITCNT	:CLEAR BIT COUNT
6266	036460	012701	007777			MOV	#256.*16.-1,R1	:LOAD COUNT
6267	036464	013737	003616	003620	20\$:	MOV	M1.BIT,M2.BIT	:SHIFT BITS
6268	036472	013737	003614	003616		MOV	PR.BIT,M1.BIT	
6269	036500	013737	003612	003614		MOV	P1.BIT,PR.BIT	
6270	036506	005037	003612			CLR	P1.BIT	
6271	036512	004737	042570			JSR	PC,WRTBIT	:WRITE ZEROS
6272	036516	104170				ERROR	170	:BIT INCORRECT
6273	036520	005237	003622			INC	BITCNT	:INCREMENT BIT COUNT
6274	036524	005301				DEC	R1	:CHECK IF READY FOR ECC
6275	036526	001356				BNE	20\$	:NO, CONTINUE
6276	036530	012701	000040			MOV	#32.,R1	:LOAD COUNT
6277	036534	042737	020000	003524		BIC	#ECCW,E.MR1	:RESET ECCW BIT
6278	036542	004737	042570		21\$:	JSR	PC,WRTBIT	:WRITE ECC FIELD
6279	036546	104170				ERROR	170	:BIT INCORRECT
6280	036550	005237	003622			INC	BITCNT	:INCREMENT COUNT
6281	036554	005301				DEC	R1	:CHECK IF READY FOR POST AMBLE
6282	036556	001371				BNE	21\$	:NO CONTINUE
6283	036560	052737	020000	003524		BIS	#ECCW,E.MR1	:SET ECCW
6284	036566	012701	000012			MOV	#10.,R1	:CHECK IF READY FOR NEXT HEADER
6285	036572	004737	042570		22\$:	JSR	PC,WRTBIT	:WRITE POSTAMBLE
6286	036576	104170				ERROR	170	:BIT INCORRECT
6287	036600	005237	003622			INC	BITCNT	:INCREMENT COUNT
6288	036604	005301				DEC	R1	:CHECK IF READY FOR NEXT HEADER
6289	036606	001371				BNE	22\$	:NO, COMPLETE POSTAMBLE
6290	036610	005237	003626			INC	SECCNT	:INCREMENT COUNT
6291	036614	005305				DEC	R5	:CHECK IF ALL HEADERS RECEIVED
6292	036616	001402				BEQ	23\$	:YES, SIMULATE INDEX
6293	036620	000137	036136			JMP	4\$	:NO GET NEXT HEADER
6294								
6295	036624	012762	000240	000026	23\$:	MOV	#DMD.MIND,RKMR1(R2)	:SIMULATE INDEX PULSE
6296	036632	012700	000004			MOV	#4,R0	
6297	036636	012762	000540	000026	25\$:	MOV	#DMD!MIND!MCLK,RKMR1(R2)	
6298	036644	012762	000240	000026		MOV	#DMD!MIND!MCLK,RKMR1(R2)	
6299	036652	005300				DEC	R0	
6300	036654	001370				BNE	25\$	
6301	036656	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6302	036664	016237	000026	003464		MOV	RKMR1(R2),T.MR1	:GET MAINT REG 1
6303	036672	012737	022040	003524		MOV	#MEWD!ECCW!DMD,E.MR1	:LOAD EXPECTED MR1
6304	036700	023737	003524	003464		CMR	E.MR1,T.MR1	:CHECK IF MR1 CORRECT
6305								: (WRITE GATE RESET)
6306	036706	001401				BEQ	28\$	:YES, CHECK IF READY SET
6307	036710	104173				ERROR	173	:MAINTENANCE REG 1 INCORRECT
6308	036712	012700	000010		28\$:	MOV	#8.,R0	:FINISH COMMAND
6309	036716	012762	000440	000026	29\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6310	036724	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6311	036732	005300				DEC	R0	
6312	036734	001370				BNE	29\$	



```

6313 036736 016237 000000 003440      MOV      RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6314 036744 012737 000226 003500      MOV      #RDY!WRHEAD<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
6315 036752 023737 003500 003440      CMP      E.CS1,T.CS1      ;CHECK IF CS1 CORRECT
6316 036760 001401          BEQ      TST45            ;;YES, GO TO NEXT TEST
6317 036762 104174          ERROR   174              ;CS1 INCORRECT
6318
6319
6320
6321
6322
6323
6324
6325
6326
6327
6328 036764 000004          TST45: SCOPE
6329 036766 012737 000010 001200      MOV      #10,$TIMES      ;;DO 10 ITERATIONS
6330 036774 013702 001270          MOV      $BASE,R2        ;LOAD RK611 BASE
6331 037000 012762 100000 000000      MOV      #CLR,RKCS1(R2) ;CLEAR RK611
6332 037006 012762 000040 000026      MOV      #DMD,RKMR1(R2) ;PUT RK611 IN DIAGNOSTIC MODE
6333 037014 012762 064472 000004      MOV      #NPRBUF,RKBA(R2) ;ISSUE READ HEADER
6334 037022 012703 064472          MOV      #NPRBUF,R3
6335 037026 012762 177676 000002      MOV      #-22.*3,RKWC(R2)
6336 037034 012762 000027 000000      MOV      #WRHEAD,RKCS1(R2)
6337 037042 012700 000312          MOV      #50.*4+2,R0     ;ISSUE CLOCKS UNTIL READY
6338                                     ; FOR INDEX PULSE
6339 037046 012762 000440 000026 1$:      MOV      #DMD!MCLK,RKMR1(R2)
6340 037054 012762 000040 000026          MOV      #DMD,RKMR1(R2)
6341 037062 005300          DEC      R0
6342 037064 001370          BNE     1$
6343 037066 012700 000004          MOV      #4,R0           ;ISSUE INDEX PULSE
6344 037072 012762 000240 000026          MOV      #DMD.MIND,RKMR1(R2)
6345 037100 012762 000640 000026 2$:      MOV      #DMD!MIND.MCLK,RKMR1(R2)
6346 037106 012762 000240 000026          MOV      #DMD!MIND,RKMR1(R2)
6347 037114 005300          DEC      R0
6348 037116 001370          BNE     2$
6349 037120 012762 000040 000026          MOV      #DMD,RKMR1(R2)
6350 037126 012700 000010          MOV      #8,R0           ;WAIT FOR WRITE GATE
6351 037132 012762 000440 000026 3$:      MOV      #DMD!MCLK,RKMR1(R2)
6352 037140 012762 000040 000026          MOV      #DMD,RKMR1(R2)
6353 037146 005300          DEC      R0
6354 037150 001370          BNE     3$
6355 037152 005037 003626          CLR      SECCNT         ;CLEAR SECTOR COUNT
6356 037156 012705 000026          MOV      #22,R5         ;LOAD NUMBER OF HEADERS
6357 037162 012762 000140 000026 4$:      MOV      #DMD.MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
6358 037170 012762 000040 000026          MOV      #DMD,RKMR1(R2)
6359 037176 012737 060475 003170          MOV      #EM233,EMW     ;LOAD ERROR MESSAGE
6360 037204 012737 062040 003524          MOV      #DMD.MEWD!ECCW.WRTGAT,E.MR1 ;INITIALIZE EXPECTED
6361                                     ; MAINT REG 1
6362 037212 005037 003612          CLR      P1.BIT
6363 037216 005037 003614          CLR      PR.BIT
6364 037222 005037 003616          CLR      M1.BIT
6365 037226 005037 003620          CLR      M2.BIT
6366
6367 037232 012700 000400          MOV      #256,R0        ;SIMULATE SYNCH
6368 037236 005037 003622          CLR      BITCNT        ;INITIALIZE BIT COUNT
  
```

6369	037242	004737	042570		5\$:	JSR	PC,WRTBIT		:WRITE ONE BIT
6370	037246	104170				ERROR	170		:DATA INCORRECT
6371	037250	005237	003622			INC	BITCNT		
6372	037254	005300				DEC	R0		:CHECK IF READY FOR DATA
6373	037256	001371				BNE	5\$		:NO, GENERATE NEXT BIT
6374	037260	012737	000001	003612		MOV	#1,P1.BIT		:PUT IN SYNCH BIT
6375	037266	004737	042570			JSR	PC,WRTBIT		
6376	037272	104170				ERROR	170		:DATA INCORRECT
6377	037274	005037	003622			CLR	BITCNT		:INITIALIZE BIT COUNT
6378	037300	012737	060541	003170		MOV	#EM234,EMW		:LOAD ERROR MESSAGE
6379	037306	012700	000003			MOV	#3,R0		:LOAD NUMBER OF WORDS IN HEADER
6380	037312	012304			10\$:	MOV	(R3)+,R4		:GET NEXT WORD
6381	037314	012701	000020			MOV	#16.,R1		:LOAD BIT COUNT
6382	037320	013737	003616	003620	12\$:	MOV	M1.BIT,M2.BIT		:SHIFT BITS
6383	037326	013737	003614	003616		MOV	PR.BIT,M1.BIT		
6384	037334	013737	003612	003614		MOV	P1.BIT,PR.BIT		
6385	037342	006004				ROR	R4		:SHIFT IN NEXT BIT
6386	037344	103403				BCS	14\$		:CHECK IF ONE
6387	037346	005037	003612			CLR	P1.BIT		:ZERO
6388	037352	000403				BR	15\$		:CLOCK IN BIT
6389									
6390	037354	012737	000001	003612	14\$:	MOV	#1,P1.BIT		:ONE
6391	037362	004737	042570		15\$:	JSR	PC,WRTBIT		:WRITE BIT
6392	037366	104170				ERROR	170		:BIT INCORRECT
6393	037370	005237	003622			INC	BITCNT		:INCREMENT BIT COUNT
6394	037374	005301				DEC	R1		:CHECK IF WORD FINISHED
6395	037376	001350				BNE	12\$		:NO, CONTINUE
6396	037400	005300				DEC	R0		:CHECK IF HEADER COMPLETE
6397	037402	001343				BNE	10\$		:NO, GET NEXT WORD
6398	037404	012737	060773	003170		MOV	#EM237,EMW		:LOAD ERROR MESSAGE
6399	037412	012701	070477			MOV	#64.+255.,R1		:LOAD COUNT
6400	037416	013737	003616	003620	18\$:	MOV	M1.BIT,M2.BIT		:SHIFT BITS
6401	037424	013737	003614	003616		MOV	PR.BIT,M1.BIT		
6402	037432	013737	003612	003614		MOV	P1.BIT,PR.BIT		
6403	037440	005037	003612			CLR	P1.BIT		
6404	037444	004737	042570			JSR	PC,WRTBIT		:WRITE ZERO
6405	037450	104170				ERROR	170		:BIT INCORRECT
6406	037452	005301				DEC	R1		:CHECK IF READY FOR DATA
6407	037454	001360				BNE	18\$		:NO, CONTINUE
6408	037456	012737	000001	003612		MOV	#1,P1.BIT		:SET SYNCH BIT
6409	037464	004737	042570			JSR	PC,WRTBIT		:WRITE SYNCH BIT
6410	037470	104170				ERROR	170		:SYNCH BIT INCORRECT
6411	037472	012737	061041	003170		MOV	#EM238,EMW		:LOAD ERROR MESSAGE
6412	037500	005037	003622			CLR	BITCNT		:CLEAR BIT COUNT
6413	037504	012701	007777			MOV	#256.*16.-1,R1		:LOAD COUNT
6414	037510	013737	003616	003620	20\$:	MOV	M1.BIT,M2.BIT		:SHIFT BITS
6415	037516	013737	003614	003616		MOV	PR.BIT,M1.BIT		
6416	037524	013737	003612	003614		MOV	P1.BIT,PR.BIT		
6417	037532	005037	003612			CLR	P1.BIT		
6418	037536	004737	042570			JSR	PC,WRTBIT		:WRITE ZEROS
6419	037542	104170				ERROR	170		:BIT INCORRECT
6420	037544	005237	003622			INC	BITCNT		:INCREMENT BIT COUNT
6421	037550	005301				DEC	R1		:CHECK IF READY FOR ECC
6422	037552	001356				BNE	20\$		:NO, CONTINUE
6423	037554	012701	000040			MOV	#32.,R1		:LOAD COUNT
6424	037560	042737	020000	003524		BIC	#ECCW,E.MR1		:RESET ECCW BIT

```

6425 037566 004737 042570      21$: JSR    PC,WRTBIT      ;WRITE ECC FIELD
6426 037572 104170              ERROR  170      ;BIT INCORRECT
6427 037574 005237 003622      INC    BITCNT      ;INCREMENT COUNT
6428 037600 005301              DEC    R1          ;CHECK IF READY FOR POST AMBLE
6429 037602 001371              BNE   21$         ;NO CONTINUE
6430 037604 052737 020000 003524  BIS   #ECCW,E.MR1  ;SET ECCW
6431 037612 012701 000012      MOV   #10.,R1     ;CHECK IF READY FOR NEXT HEADER
6432 037616 004737 042570      22$: JSR    PC,WRTBIT      ;WRITE POSTAMBLE
6433 037622 104170              ERROR  170      ;BIT INCORRECT
6434 037624 005237 003622      INC    BITCNT      ;INCREMENT COUNT
6435 037630 005301              DEL   R1          ;CHECK IF READY FOR NEXT HEADER
6436 037632 001371              BNE   22$         ;NO, COMPLETE POSTAMBLE
6437 037634 005237 003626      INC    SECCNT     ;INCREMENT COUNT
6438 037640 005305              DEC   R5          ;CHECK IF ALL HEADERS RECEIVED
6439 037642 001402              BEQ   23$         ;YES, SIMULATE INDEX
6440 037644 000137 037162      JMP   4$         ;NO GET NEXT HEADER
6441
6442 037650 012762 000240 000026 23$: MOV   #DMD!MIND,RKMR1(R2) ;SIMULATE INDEX PULSE
6443 037656 012700 000004      MOV   #4,R0
6444 037662 012762 000640 000026 25$: MOV   #DMD!MIND!MCLK,RKMR1(R2)
6445 037670 012762 000240 000026  MOV   #DMD!MIND,RKMR1(R2)
6446 037676 005300              DEC   R0
6447 037700 001370              BNE   25$
6448 037702 012762 000040 000026  MOV   #DMD,RKMR1(R2)
6449 037710 016237 000026 003464  MOV   RKMR1(R2),T.MR1 ;GET MAINT REG 1
6450 037716 012737 022040 003524  MOV   #MEWD!ECCW.DMD,E.MR1 ;LOAD EXPECTED MR1
6451 037724 023737 003524 003464  CMP   E.MR1,T.MR1 ;CHECK IF MR1 CORRECT
6452                                ; (WRITE GATE RESET)
6453                                BEQ   28$         ;YES, CHECK IF READY SET
6454 037734 104173              ERROR  173      ;MAINTENANCE REG 1 INCORRECT
6455 037736 012700 000010      28$: MOV   #8.,R0
6456 037742 012762 000440 000026 29$: MOV   #DMD!MCLK,RKMR1(R2)
6457 037750 012762 000040 000026  MOV   #DMD,RKMR1(R2)
6458 037756 005300              DEC   R0
6459 037760 001370              BNE   29$
6460 037762 016237 000000 003440  MOV   RK(CS1(R2),T.CS1 ;GET COMMAND AND STATUS REG 1
6461 037770 012737 000226 003500  MOV   #RDY!WRHEAD&<^C<GO>>,F.CS1 ;LOAD EXPECTED CS1
6462 037776 023737 003500 003440  CMP   E.CS1,T.CS1 ;CHECK IF CS1 CORRECT
6463 040004 001401              BEQ   TST46      ;:YES, GO TO NEXT TEST
6464 040006 104174              ERROR  174      ;CS1 INCORRECT
6465
6466
6467
6468
6469
6470
6471
6472
6473
6474
6475
6476
6477
6478
6479 040010 000004
6480 040012 012737 000144 001200

```

```

*****
*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.
*****
TST46:  SCOPE
        MOV   #100.,$TIMES      ;;DO 100. ITERATIONS

```

ZR6CE0 RK611 DSKLS CTRL PR3  
ZR6CE.P11 27-AUG-81 10:24

MACV11 30(1046)  
T46

28-AUG-81 10:35 PAGE 121  
WRITE HEADER IN 24 SECTOR FORMAT

SEQ 0120

```

6481 040020 013702 001270      MOV      $BASE,R2          ;LOAD RK611 BASE
6482 040024 012762 100000 000000  MOV      #CLR,RKCS1(R2)    ;CLEAR RK611
6483 040032 012762 000040 000026  MOV      #DMD,RKMR1(R2)   ;PUT RK611 TO DIAGNOSTIC MODE
6484 040040 012762 064412 000004  MOV      #HEAD1,RKBA(R2)  ;ISSUE WRITE HEADER
6485 040046 012703 064412      MOV      #HEAD1,R3
6486 040052 012762 177772 000002  MOV      #-6,RKWC(R2)
6487 040060 012762 010027 000000  MOV      #CFMT!WRHEAD,RKCS1(R2)
6488 040066 012700 000312      MOV      #50.*4+2,R0      ;ISSUE ENOUGH CLOCKS UNTIL
6489                                     ; READY FOR INDEX PULSE
6490 040072 012762 000440 000026  8:      MOV      #DMD!MCLK,RKMR1(R2)
6491 040100 012762 000040 000026  MOV      #DMD,RKMR1(R2)
6492 040106 005300      DEC      R0
6493 040110 001370      BNE     1$
6494 040112 012700 000004      MOV      #4,R0           ;ISSUE INDEX PULSE
6495 040116 012762 000240 000026  MOV      #DMD!MIND,RKMR1(R2)
6496 040124 012762 000640 000026  2$:      MOV      #DMD.MIND!MCLK,RKMR1(R2)
6497 040132 012762 000240 000026  MOV      #DMD!MIND,RKMR1(R2)
6498 040140 005300      DEC      R0
6499 040142 001370      BNE     2$
6500 040144 005037 003626      CLR      SECCNT          ;CLEAR SECTOR COUNT
6501 040150 012762 000040 000026  MOV      #DMD,RKMR1(R2)
6502 040156 012705 000002      MOV      #2,R5           ;LOAD NUMBER OF HEADERS
6503 040162 012700 000010      MOV      #8,R0           ;WAIT FOR WRITE GATE
6504 040166 012762 000440 000026  3$:      MOV      #DMD!MCLK,RKMR1(R2)
6505 040174 012762 000040 000026  MOV      #DMD,RKMR1(R2)
6506 040202 005300      DEC      R0
6507 040204 001370      BNE     3$
6508 040206 012762 000140 000026  4$:      MOV      #DMD!MSP,RKMR1(R2) ;SIMULATE SECTOR PULSE
6509 040214 012762 000040 000026  MOV      #DMD,RKMR1(R2)
6510 040222 012737 061100 003170  MOV      #EM239,EMW      ;LOAD ERROR MESSAGE
6511 040230 012737 062040 003524  MOV      #DMD.MEWD!ECCW!WRIGAT,E.MR1 ;INITIALIZE EXPECTED
6512                                     ; MAINT REG 1
6513 040236 005037 003612      CLR      P1.BIT
6514 040242 005037 003614      CLR      PR.BIT
6515 040246 005037 003616      CLR      M1.BIT
6516 040252 005037 003620      CLR      M2.BIT
6517 040256 012700 000400      MOV      #256,R0         ;SIMULATE SYNCH
6518 040262 005037 003622      CLR      BITCNT          ;INITIALIZE BIT COUNT
6519 040266 004737 042570 5$:      JSR     PC,WRITBIT      ;WRITE ONE BIT
6520 040272 104170      ERROR   170             ;DATA INCORRECT
6521 040274 005237 003622      INC      BITCNT
6522 040300 005300      DEC      R0             ;CHECK IF READY FOR DATA
6523 040302 001371      BNE     5$             ;NO,GENERATE NEXT BIT
6524 040304 012737 000001 003612  MOV      #1,P1.BIT      ;PUT IN SYNCH BIT
6525 040312 004737 042570      JSR     PC,WRITBIT
6526 040316 104170      ERROR   170             ;DATA INCORRECT
6527 040320 005037 003622      CLR      BITCNT          ;INITIALIZE BIT COUNT
6528 040324 012737 061172 003170  MOV      #EM240,EMW      ;LOAD ERROR MESSAGE
6529 040332 012700 000003      MOV      #3,R0           ;LOAD NUMBER OF WORDS IN HEADER
6530 040336 012304 000000 10$:      MOV      (R3)+,R4        ;GET NEXT WORD
6531 040340 012701 000020      MOV      #16,R1         ;LOAD BIT COUNT
6532 040344 013737 003616 003620  12$:      MOV      M1.BIT,M2.BIT  ;SHIFT BITS
6533 040352 013737 003614 003616  MOV      PR.BIT,M1.BIT
6534 040360 013737 003612 003614  MOV      P1.BIT,PR.BIT
6535 040366 006004      ROR     R4             ;SHIFT IN NEXT BIT
6536 040370 103403      BCS    14$            ;CHECK IF ONE

```

```

6537 040372 005037 003612 CLR P1.BIT :ZERO
6538 040376 000403 BR 15$ :CLOCK IN BIT
6539
6540 040400 012737 000001 003612 14$: MOV #1,P1.BIT :ONE
6541 040406 004737 042570 15$: JSR PC,WRTBIT :WRITE BIT
6542 040412 104170 ERROR 170 :BIT INCORRECT
6543 040414 005237 003622 INC BITCNT :INCREMENT BIT COUNT
6544 040420 005301 DEC R1 :CHECK IF WORD FINISHED
6545 040422 001350 RNE 12$ :NO, CONTINUE
6546 040424 005300 DEC R0 :CHECK IF HEADER COMPLETE
6547 040426 001343 BNE 10$ :NO, GET NEXT WORD
6548 040430 012701 000020 MOV #16,R1 :LOAD BIT COUNT FOR NEXT WORD
6549 040434 013737 003616 003620 18$: MOV M1.BIT,M2.BIT :SHIFT BITS
6550 040442 013737 003614 003616 MOV PR.BIT,M1.BIT
6551 040450 013737 003612 003614 MOV P1.BIT,PR.BIT
6552 040456 005037 003612 CLR P1.BIT
6553 040462 004737 042570 JSR PC,WRTBIT :WRITE ZERO
6554 040466 104170 ERROR 170 :BIT INCORRECT
6555 040470 005237 003622 INC BITCNT :INCREMENT
6556 040474 005301 DEC R1 :CHECK IF READY FOR NEXT HEADER
6557 040476 001356 BNE 18$ :NO, CONTINUE
6558 040500 005237 003626 INC SECCNT :INCREMENT
6559 040504 005305 DEC R5 :CHECK IF SECOND HEADER WRITTEN
6560 040506 001237 BNF 4$ :NO, DO SECOND HEADER
6561 040510 012762 000240 000026 MOV #DMD.MIND,RKMR1(R2) :SIMULATE INDEX PULSE
6562 040516 012700 000004 MOV #4,R0
6563 040522 012762 000640 000026 20$: MOV #DMD!MIND.MCLK,RKMR1(R2)
6564 040530 012762 000240 000026 MOV #DMD.MIND,RKMR1(R2)
6565 040536 005300 DEC R0
6566 040540 001370 BNE 20$
6567 040542 012762 000040 000026 MOV #DMD,RKMR1(R2)
6568 040550 016237 000026 003464 MOV RKMR1(R2),T.MR1 :GET MAINT REG 1
6569 040556 012737 022040 003524 MOV #MEWD.ECCW!DMD,E.MR1 :LOAD EXPECTED MR1
6570 040564 023737 003524 003464 CMP E.MR1,T.MR1 :CHECK MR1 CORRECT (WRITE GATE RESET)
6571 040572 001401 BEQ 25$ :YES, CHECK IF READY SET
6572 040574 104173 ERROR 173 :MAINT REG 1 INCORRECT
6573 040576 012700 000010 25$: MOV #8,R0 :FINISH COMMAND
6574 040602 012762 000440 000026 26$: MOV #DMD!MCLK,RKMR1(R2)
6575 040610 012762 000040 000026 MOV #DMD,RKMR1(R2)
6576 040616 005300 DEC R0
6577 040620 001370 BNE 26$
6578 040622 016237 000000 003440 MOV RKCS1(R2),T.CS1 :GET COMMAND AND STATUS REG 1
6579 040630 012737 010226 003500 MOV #RDY!CFMT!WRHEAD<^C<GO>>,E.CS1 :LOAD EXPECTED CS1
6580 040636 023737 003500 003440 CMP E.CS1,T.CS1 :CHECK IF CS1 CORRECT
6581 040644 001401 BEQ TST47 :;YES, GO ON TO NEXT TEST
6582 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

```

6583  
6584  
6585  
6586  
6587  
6588  
6589  
6590  
6591  
6592

```

6593          : *      MAINTENANCE MODE. MAKE SURE FORMAT ERROR,
6594          : *      DRIVE AVAILABLE AND CONTROLLER ERROR SET.
6595          : *
6596          : *
6597          : *****
6597 040650 000004 TST47: SCOPE
6598 040652 012737 000144 001200 MOV #100,$TIMES ;;DO 100. ITERATIONS
6599 040660 013702 001270 MOV $BASE,R2 ;:LOAD RK611 BASE
6600 040664 012762 000040 000010 MOV #SCLR,RKCS2(R2) ;:CLEAR RK611 SUBSYSTEM
6601 040672 012762 000040 000026 MOV #DMD,RKMR1(R2) ;:PUT RK611 IN MAINTENANCE MODE
6602 040700 012762 000043 000020 MOV #43,RKDCYL(R2) ;:LOAD CYLINDER ADDRESS REG
6603 040706 012762 000025 000000 MOV #RDHEAD,RKCS1(R2) ;:ISSUE READ HEADER
6604 040714 012700 000132 MOV #22,*4+2,R0 ;:ISSUE CLOCKS UNTIL PHASE ADDRESS 6
6605 040720 012762 000440 000026 1$: MOV #DMD!MCLK,RKMR1(R2)
6606 040726 012762 000040 000026 MOV #DMD,RKMR1(R2)
6607 040734 005300 DEC R0
6608 040736 001370 BNE 1$
6609 040740 005062 000026 CLR RKMR1(R2) ;:FINISH COMMAND IN NORMAL MODE
6610 040744 013700 003632 MOV WAITIM,R0 ;:WAIT FOR READY
6611 040750 105762 000000 2$: TSTB RKCS1(R2)
6612 040754 100402 BMI 3$
6613 040756 005300 DFC R0
6614 040760 001373 BNE 2$
6615 040762 016237 000000 003440 3$: MOV RKCS1(R2),T.CS1 ;:STORE COMMAND AND STATUS REG 1
6616 040770 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;:STORE COMMAND AND STATUS REG 2
6617 040776 016237 000012 003452 MOV RKDS(R2),T.DS ;:STORE DRIVE STATUS REG
6618 041004 016237 000014 003454 MOV RKER(R2),T.ER ;:STORE ERROR REG
6619 041012 012737 100224 003500 MOV #CERR!RDY!RDHEAD<^C<GO>>,E.CS1 ;:LOAD EXPECTED CS1
6620 041020 012737 000100 003510 MOV #IR,E.CS2 ;:LOAD EXPECTED CS2
6621 041026 012737 100001 003512 MOV #SVAL!DRA,E.DS ;:LOAD EXPECTED DRIVE STATUS REG
6622 041034 012737 000020 003514 MOV #FMTE,E.ER ;:LOAD EXPECTED ERROR REG
6623 041042 023737 003500 003440 CMP E.CS1,T.CS1 ;:CHECK COMMAND AND STATUS REG 1 CORRECT
6624 041050 001401 BEQ 4$ ;:YES, CONTINUE
6625 041052 104175 ERROR 175
6626 041054 023737 003510 003450 4$: CMP E.CS2,T.CS2 ;:CHECK COMMAND AND STATUS REG 2 CORRECT
6627 041062 001401 BEQ 5$ ;:YES, CONTINUE
6628 041064 104176 ERROR 176
6629 041066 023737 003512 003452 5$: CMP E.DS,T.DS ;:CHECK IF DRIVE STATUS REG CORRECT
6630 041074 001401 BEQ 6$ ;:YES, CONTINUE
6631 041076 104177 ERROR 177
6632 041100 023737 003514 003454 6$: CMP E.ER,T.ER ;:CHECK IF ERR REG CORRECT
6633 041106 001401 BEQ 7$ ;:YES, CONTINUE
6634 041110 104200 ERROR 200
6635 041112 013737 003440 003540 7$: MOV T.CS1,P.CS1 ;:STORE PREVIOUS CONTENTS OF
6636 041120 013737 003450 003550 MOV T.CS2,P.CS2 ;: COMMAND AND STATUS REG 1
6637 041126 013737 003452 003552 MOV T.DS,P.DS ;: COMMAND AND STATUS REG 2
6638 041134 013737 003454 003554 MOV T.ER,P.ER ;: DRIVE STATUS REG
6639          : AND ERROR REG
6640 041142 012762 100000 000000 MOV #CCLR,RKCS1(R2) ;:CLEAR RK611
6641 041150 016237 000000 003440 MOV RKCS1(R2),T.CS1 ;:STORE COMMAND AND STATUS REG 1
6642 041156 016237 000010 003450 MOV RKCS2(R2),T.CS2 ;:STORE COMMAND AND STATUS REG 2
6643 041164 016237 000012 003452 MOV RKDS(R2),T.DS ;:STORE DRIVE STATUS REG
6644 041172 016237 000014 003454 MOV RKER(R2),T.ER ;:STORE ERROR REG
6645 041200 012737 000200 003500 MOV #RDY,E.CS1 ;:LOAD EXPECTED CS1
6646 041206 012737 000100 003510 MOV #IR,E.CS2 ;:LOAD EXPECTED CS2
6647 041214 005037 003512 CLR E.DS ;:LOAD EXPECTED DRIVE STATUS REG
6648 041220 005037 003514 CLR E.ER ;:LOAD EXPECTED ERROR REG

```

```

6649 041224 023737 003500 003440      CMP      E.CS1,T.CS1      ;CHECK CS1 CORRECT
6650 041232 001401          BEQ      11$             ;YES, CONTINUE
6651 041234 104211          ERROR    211            ;CS1 INCORRECT
6652 041236 023737 003510 003450 11$:  CMP      E.CS2,T.CS2      ;CHECK CS2 CORRECT
6653 041244 001401          BEQ      12$             ;YES, CONTINUE
6654 041246 104212          ERROR    212            ;CS2 INCORRECT
6655 041250 023737 003512 003452 12$:  CMP      E.DS,T.DS      ;CHECK DRIVE STATUS CORRECT
6656 041256 001401          BEQ      13$             ;YES, CONTINUE
6657 041260 104213          ERROR    213            ;DRIVE STATUS REG INCORRECT
6658 041262 023737 003514 003454 13$:  CMP      E.ER,T.ER      ;CHECK IF ERROR REG CORRECT
6659 041270 001401          BEQ      14$             ;YES, GO ON TO NEXT TEST
6660 041272 104214          ERROR    214            ;ERROR REG INCORRECT
6661 041274          14$:
6662
6663
6664
6665
6666
6667
6668
6669
6670
6671
6672
6673
6674 041274 000004          *****
6675 041276 012737 000144 001200 1$:  TEST 50      FORMAT ERROR (PART 2)
6676 041304 013702 001270          *
6677 041310 012762 000040 000010          * CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
6678 041316 012762 000040 000026          * CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ HEADER TO AN
6679 041324 012762 000003 000020          * RK06, IN 24 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE 0.
6680 041332 012762 010025 000000          * CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6. TURN OFF
6681 041340 012700 000132          * MAINTENANCE MODE. MAKE SURE FORMAT ERROR,
6682 041344 012762 000440 000026 1$:  * DRIVE AVAILABLE AND CONTROLLER ERROR SET.
6683 041352 012762 000040 000026          *
6684 041360 005300          *****
6685 041362 001370          TST50:  SCOPE
6686 041364 005062 000026          MOV      #100, $TIMES    ;;DO 100. ITERATIONS
6687 041370 013700 003632          MOV      $BASE,R2       ;LOAD RK611 BASE
6688 041374 105762 000000 2$:  MOV      #SCLR,RKCS2(R2) ;CLEAR RK611 SUBSYSTEM
6689 041400 100402          MOV      #DMD,RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
6690 041402 005300          MOV      #3,RKDCYL(R2)  ;LOAD CYLINDER ADDRESS REG
6691 041404 001373          MOV      #RDHEAD!CFMT,RKCS1(R2) ;ISSUE READ HEADER
6692 041406 016237 000000 003440 3$:  MOV      #22,*4+2,R0     ;ISSUE CLOCKS UNTIL PHASE ADDRESS 6
6693 041414 016237 000010 003450          MOV      #DMD!MCLK,RKMR1(R2)
6694 041422 016237 000012 003452          MOV      #DMD,RKMR1(R2)
6695 041430 016237 000014 003454          DEC      R0
6696 041436 012737 110224 003500          BNE     2$
6697 041444 012737 000100 003510          CLR     RKMR1(R2)       ;FINISH COMMAND IN NORMAL MODE
6698 041452 012737 100001 003512          MOV     WAITIM,R0      ;WAIT FOR READY
6699 041460 012737 000030 003514          TSTB   RKCS1(R2)
6700 041466 023737 003500 003440          BMI     3$
6701 041474 001401          DEC     R0
6702 041476 104201          BNE     2$
6703 041500 023737 003510 003450 4$:  MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
6704 041506 001401          MOV     RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
        MOV     RKDS(R2),T.DS ;STORE DRIVE STATUS REG
        MOV     RKER(R2),T.ER ;STORE ERROR REG
        MOV     #CERR!RDY!RDHEAD.CFMT<^C<GO>>,E.CS1 ;LOAD EXPECTED CS1
        MOV     #IR,E.CS2 ;LOAD EXPECTED CS2
        MOV     #SVAL!DRA,E.DS ;LOAD EXPECTED DRIVE STATUS REG
        MOV     #FMTE!DRPAR,E.ER ;LOAD EXPECTED ERROR REG
        CMP     E.CS1,T.CS1 ;CHECK COMMAND AND STATUS REG 1 CORRECT
        BEQ     4$ ;YES, CONTINUE
        ERROR 201
        CMP     E.CS2,T.CS2 ;CHECK COMMAND AND STATUS REG 2 CORRECT
        BEQ     5$ ;YES, CONTINUE
  
```



```

6705 041510 104202          ERROR 202
6706 041512 023737 003512 003452 5$: CMP E.DS,T.DS ;CHECK IF DRIVE STRATUS REG CORRECT
6707 041520 001401          BEQ 6$ ;YES, CONTINUE
6708 041522 104203          ERROR 203
6709 041524 023737 003514 003454 6$: CMP E.ER,T.ER ;CHECK IF ERR REG CORRECT
6710 041532 001401          BEQ 7$ ;YES, CONTINUE
6711 041534 104204          ERROR 204
6712 041536 013737 003440 003540 7$: MOV T.CS1,P.CS1 ;STORE PREVIOUS CONTENTS OF
6713 041544 013737 003450 003550      MOV T.CS2,P.CS2 ; COMMAND AND STATUS REG 1
6714 041552 013737 003452 003552      MOV T.DS,P.DS ; COMMAND AND STATUS REG 2
6715 041560 013737 003454 003554      MOV T.ER,P.ER ; DRIVE STATUS REG
6716          ; AND ERROR REG
6717 041566 012762 100000 000000      MOV #CLR,RKCS1(R2) ;CLEAR RK611
6718 041574 016237 000000 003440      MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REG 1
6719 041602 016237 000010 003450      MOV RKCS2(R2),T.CS2 ;STORE COMMAND AND STATUS REG 2
6720 041610 016237 000012 003452      MOV RKDS(R2),T.DS ;STORE DRIVE STATUS REG
6721 041616 016237 000014 003454      MOV RKER(R2),T.ER ;STORE ERROR REG
6722 041624 012737 000200 003500      MOV #RDY,E.CS1 ;LOAD EXPECTED CS1
6723 041632 012737 000100 003510      MOV #IR,E.CS2 ;LOAD EXPECTED CS2
6724 041640 005037 003512          CLR E.DS ;LOAD EXPECTED DRIVE STATUS REG
6725 041644 005037 003514          CLR E.ER ;LOAD EXPECTED ERROR REG
6726 041650 023737 003500 003440      CMP E.CS1,T.CS1 ;CHECK CS1 CORRECT
6727 041656 001401          BEQ 11$ ;YES, CONTINUE
6728 041660 104211          ERROR 211 ;CS1 INCORRECT
6729 041662 023737 003510 003450 11$: CMP E.CS2,T.CS2 ;CHECK CS2 CORRECT
6730 041670 001401          BEQ 12$ ;YES, CONTINUE
6731 041672 104212          ERROR 212 ;CS2 INCORRECT
6732 041674 023737 003512 003452 12$: CMP E.DS,T.DS ;CHECK DRIVE STATUS CORRECT
6733 041702 001401          BEQ 13$ ;YES, CONTINUE
6734 041704 104213          ERROR 213 ;DRIVE STATUS REG INCORRECT
6735 041706 023737 003514 003454 13$: CMP E.ER,T.ER ;CHECK IF ERROR REG CORRECT
6736 041714 001401          BEQ 14$ ;YES, GO ON TO NEXT TEST
6737 041716 104214          ERROR 214 ;ERROR REG INCORRECT
6738 041720          14$:

```

```

6739
6740
6741 :*****
6742 :*TEST 51 FAULT SETTING CONTROLLER ERROR
6743 :*
6744 :* CLEAR THE RK06 SUBSYSTEM WITH A SUBSYSTEM CLEAR. PUT THE
6745 :* CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE HEADER TO
6746 :* AN RK06, IN 26 SECTOR FORMAT, CYLINDER 3, HEAD 0, DRIVE
6747 :* 0. CLOCK IN MAINTENANCE MODE UNTIL PHASE ADDRESS 6.
6748 :* TURN OFF MAINTENANCE MODE. MAKE SURE DRIVE
6749 :* AVAILABLE AND CONTROLLER ERROR SET.
6750 :*****

```

```

6751 041720 000004          TST51: SCOPE
6752 041722 012737 000144 001200      MOV #100,$TIMES ;DO 100. ITERATIONS
6753 041730 013702 001270          MOV $BASE,R2 ;LOAD RK611 BASE
6754 041734 012762 000040 000010      MOV #CLR,RKCS2(R2) ;CLEAR RK611 SUBSYSTEM
6755 041742 012762 000040 000026      MOV #DMD,RKMR1(R2) ;PUT RK611 IN MAINTENANCE MODE
6756 041750 012762 000003 000020      MOV #3,RKDCYL(R2) ;LOAD CYLINDER ADDRESS REG
6757 041756 012762 177775 000002      MOV #-3,RKWC(R2) ;LOAD WORD COUNT
6758 041764 012762 064714 000004      MOV #WRBJFF,RKBA(R2) ;LOAD BUS ADDRESS
6759 041772 012762 000027 000000      MOV #WRHEAD,RKCS1(R2) ;ISSUE WRITE HEADER
6760 042000 012700 000132          MOV #22.*4+2,R0 ;ISSUE CLOCKS UNTIL PHASE ADDRESS 6

```



6761	042004	012762	000440	000026	1\$:	MOV	#DMD!MCLK,RKMR1(R2)	
6762	042012	012762	000040	000026		MOV	#DMD,RKMR1(R2)	
6763	042020	005300				DEC	R0	
6764	042022	001370				BNE	1\$	
6765	042024	005062	000026			CLR	RKMR1(R2)	:FINISH COMMAND IN NORMAL MODE
6766	042030	013700	003632			MOV	WAITIM,R0	:WAIT FOR READY
6767	042034	105762	000000		2\$:	TSTB	RKCS1(R2)	
6768	042040	100402				BMI	3\$	
6769	042042	005300				DEC	R0	
6770	042044	001373				BNE	2\$	
6771	042046	016237	000000	003440	3\$:	MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG 1
6772	042054	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG 2
6773	042062	016237	000012	003452		MOV	RKDS(R2),T.DS	:STORE DRIVE STATUS REG
6774	042070	016237	000014	003454		MOV	RKER(R2),T.ER	:STORE ERROR REG
6775	042076	012737	100226	003500		MOV	#CERR!RDY!WRHEAD<^C<GO>>,E.CS1	:LOAD EXPECTED CS1
6776	042104	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
6777	042112	012737	100001	003512		MOV	#SVAL.DRA,E.DS	:LOAD EXPECTED DRIVE STATUS REG
6778	042120	012737	000000	003514		MOV	#0,E.ER	:LOAD EXPECTED ERROR REG
6779	042126	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK COMMAND AND STATUS REG 1 CORRECT
6780	042134	001401				BEQ	4\$	:YES, CONTINUE
6781	042136	104205				ERROR	205	
6782	042140	023737	003510	003450	4\$:	CMP	E.CS2,T.CS2	:CHECK COMMAND AND STATU REG 2 CORRECT
6783	042146	001401				BEQ	5\$	:YES, CONTINUE
6784	042150	104206				ERROR	206	
6785	042152	023737	003512	003452	5\$:	CMP	E.DS,T.DS	:CHECK IF DRIVE STRATUS REG CORRECT
6786	042160	001401				BEQ	6\$	:YES, CONTINUE
6787	042162	104207				ERROR	207	
6788	042164	023737	003514	003454	6\$:	CMP	E.ER,T.ER	:CHECK IF ERR REG CORRECT
6789	042172	001401				BEQ	7\$	:YES, CONTINUE
6790	042174	104210				ERROR	210	
6791	042176	013737	003440	003540	7\$:	MOV	T.CS1,P.CS1	:STORE PREVIOUS CONTENTS OF
6792	042204	013737	003450	003550		MOV	T.CS2,P.CS2	: COMMAND AND STATUS REG 1
6793	042212	013737	003452	003552		MOV	T.DS,P.DS	: COMMAND AND STATUS REG 2
6794	042220	013737	003454	003554		MOV	T.ER,P.ER	: DRIVE STATUS REG
6795								: AND ERROR REG
6796	042226	012762	100000	000000		MOV	#CLR,RKCS1(R2)	:CLEAR RK611
6797	042234	016237	000000	003440		MOV	RKCS1(R2),T.CS1	:STORE COMMAND AND STATUS REG 1
6798	042242	016237	000010	003450		MOV	RKCS2(R2),T.CS2	:STORE COMMAND AND STATUS REG 2
6799	042250	016237	000012	003452		MOV	RKDS(R2),T.DS	:STORE DRIVE STATUS REG
6800	042256	016237	000014	003454		MOV	RKER(R2),T.ER	:STORE ERROR REG
6801	042264	012737	000200	003500		MOV	#RDY,E.CS1	:LOAD EXPECTED CS1
6802	042272	012737	000100	003510		MOV	#IR,E.CS2	:LOAD EXPECTED CS2
6803	042300	005037	003512			CLR	E.DS	:LOAD EXPECTED DRIVE STATUS REG
6804	042304	005037	003514			CLR	E.ER	:LOAD EXPECTED ERROR REG
6805	042310	023737	003500	003440		CMP	E.CS1,T.CS1	:CHECK CS1 CORRECT
6806	042316	001401				BEQ	11\$	:YES, CONTINUE
6807	042320	104211				ERROR	211	:CS1 INCORRECT
6808	042322	023737	003510	003450	11\$:	CMP	E.CS2,T.CS2	:CHECK CS2 CORRECT
6809	042330	001401				BEQ	12\$	:YES, CONTINUE
6810	042332	104212				ERROR	212	:CS2 INCORRECT
6811	042334	023737	003512	003452	12\$:	CMP	E.DS,T.DS	:CHECK DRIVE STATUS CORRECT
6812	042342	001401				BEQ	13\$	:YES, CONTINUE
6813	042344	104213				ERROR	213	:DRIVE STATUS REG INCORRECT
6814	042346	023737	003514	003454	13\$:	CMP	E.ER,T.ER	:CHECK IF ERROR REG CORRECT
6815	042354	001401				BEQ	14\$	:YES, GO ON TO NEXT TEST
6816	042356	104214				ERROR	214	:ERROR REG INCORRECT

CZR6CEO RK611 DSKLS CTRL PRT3  
CZR6CE.P11 27-AUG-81 10:24

MACV11 30(1046) 28-AUG-81 10:35 PAGE 127  
T51 FAULT SETTING CONTROLLER ERROR

J 10

SEG C126

6817 042360  
6818

148:

6819  
6820  
6821  
6822  
6823  
6824  
6825  
6826  
6827  
6828 042360  
6829 042360 000004  
6830 042362 005037 001102  
6831 042366 005037 001200  
6832 042372 005237 001222  
6833 042376 042737 100000 001222  
6834 042404 005327  
6835 042406 000001  
6836 042410 003063  
6837 042412 012737  
6838 042414 000001  
6839 042416 042406  
6840 042420 104401 042426  
6841 042424 000407  
6842  
6843 042444  
6844 042444 013746 001222  
6845  
6846 042450 104405  
6847 042452 104401 042460  
6848 042456 000421  
6849  
6850 042522  
6851 042522 013746 001112  
6852  
6853 042526 104405  
6854 042530 104401 001211  
6855 042534 005037 001112  
6856 042540 013700 000042  
6857 042544 001405  
6858 042546 000005  
6859 042550 004710  
6860 042552 000240  
6861 042554 000240  
6862 042556 000240  
6863 042560  
6864 042560 000137  
6865 042562 004636  
6866 042564 377 377 000  
6867 042570  
6868  
6869  
6870  
6871  
6872 042570 052737 002400 003524  
6873 042576 012762 000440 000026  
6874 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 $STINM          ;;ZERO THE TEST NUMBER
      CLR $TIMES         ;;ZERO THE NUMBER OF ITERATIONS
      INC $PASS          ;;INCREMENT THE PASS NUMBER
      BIC #10000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
      DEC (PC)+          ;;LOOP?
$EOPCT: .WORD 1
      BGT $DOAGN         ;;YES
      MOV (PC)+,@(PC)+   ;;RESTORE COUNTER
$ENDCT: .WORD 1
      $FOPCT
      TYPE ,65$          ;;TYPE ASCIZ STRING
      BR 64$             ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <12><15>/END PASS #/
64$:
      MOV $PASS,-(SP)    ;;SAVE $PASS FOR TYPEOUT
                          ;;TYPE PASS NUMBER
                          ;;GO TYPE--DECIMAL ASCII WITH SIGN
      TYPDS
      TYPE ,67$          ;;TYPE ASCIZ STRING
      BR 66$             ;;GET OVER THE ASCIZ
;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
66$:
      MOV $ERTTL,-(SP)   ;;SAVE $ERTTL FOR TYPEOUT
                          ;;TOTAL NUMBER OF ERRORS
                          ;;GO TYPE--DECIMAL ASCII WITH SIGN
      TYPDS
      TYPE ,$CRLF        ;;TYPE CARRIAGE RETURN, LINE FEED
      CLR $ERTTL        ;;CLEAR ERROR TOTAL
$GET42: MOV @#42,R0      ;;GET MONITOR ADDRESS
      BEQ $DOAGN        ;;BRANCH IF NO MONITOR
      RESET            ;;CLEAR THE WORLD
$ENDAD: JSR PC,(R0)     ;;GO TO MONITOR
      NOP              ;;SAVE ROOM
      NOP              ;;FOR
      NOP              ;;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
WRITBIT: 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
```

```

6875 042612 023737 003524 003464      CMP      E.MR1,T.MR1      ;CHECK IF MAINT REG 1 CORRECT
6876 042620 001416                      BEQ      3$              ;YES, PROVIDE DOWNWARD TRANSITION
6877 042622 012737 042642 001202      MOV      #1$,$ESCAPE    ;LOAD ESCAPE FOR LOOP ON ERROR
6878 042630 012737 064033 003172      MOV      #EMW1,EMW+2    ;LOAD ERROR MESSAGE
6879 042636 011646                      MOV      (SP),-(SP)     ;SAVE RETURN
6880 042640 000207                      RTS      PC              ;MR1 INCORRECT ON UPWARD TRANSITION
6881
6882 042642 032777 001000 136270 1$:      BIT      #SW9,@SWR      ;CHECK IF LOOP ON ERROR
6883 042650 001402                      BEQ      3$              ;NO, CONTINUE
6884 042652 000137 043524                      JMP      63$            ;YES, LOOP ON ERROR
6885
6886 042656 042737 014400 003524 3$:      BIC      #MCLK.PCA!PCD,E.MR1 ;INITIALIZE MAINTENANCE REG. 1
6887 042664 052737 042000 003524      BIS      #MEWD!WRTGAT,E.MR1
6888 042672 005737 003614                      TST      PR.BIT         ;CHECK IF ONE
6889 042676 001152                      BNE      20$            ;YES, SIMULATE ONE
6890 042700 005737 003616                      TST      M1.BIT         ;CHECK IF PREVIOUS ONE
6891 042704 001023                      BNE      10$            ;YES, NO TRANSITION
6892 042706 042737 002000 003524      BIC      #MEWD,E.MR1    ;INDICATE TRANSITION
6893 042714 005737 003612                      TST      P1.BIT         ;CHECK IF NEXT BIT = 1
6894 042720 001007                      BNE      5$             ;YES, CHECK FOR PRECOMP ADVANCE
6895 042722 005737 003620                      TST      M2.BIT         ;CHECK FOR PRECOMP. ADVANCE
6896 042726 001412                      BEQ      10$            ;NO, CLOCK IN ZERO
6897 042730 052737 010000 003524      BIS      #PCD,E.MR1     ;SET PRECOMP. DELAY
6898 042736 000406                      BR       10$            ;CLOCK IN ZERO
6899
6900 042740 005737 003620                      5$:      TST      M2.BIT         ;CHECK FOR PRECOMP. ADVANCE
6901 042744 001003                      BNE      10$            ;CLOCK IN ZERO
6902 042746 052737 004000 003524      BIS      #PCA,E.MR1     ;SET PRECOMP. ADVANCE
6903 042754 012762 000040 000026 10$:      MOV      #DMD,RKMR1(R2) ;CLOCK IN DATA BIT
6904 042762 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MR1
6905 042770 023737 003464 003524      CMP      T.MR1,E.MR1    ;CHECK IF MR1 CORRECT
6906 042776 001416                      BEQ      12$            ;YES, CONTINUE
6907 043000 012737 043020 001202      MOV      #11$,$ESCAPE   ;LOAD ESCAPE FOR LOOP ON ERROR
6908 043006 012737 064122 003172      MOV      #EMW2,EMW+2    ;LOAD ERROR MESSAGE
6909 043014 011646                      MOV      (SP),-(SP)     ;SAVE RETURN
6910 043016 000207                      RTS      PC              ;MR1 INCORRECT
6911
6912 043020 032777 001000 136112 11$:      BIT      #SW9,@SWR      ;CHECK IF LOOP ON ERROR
6913 043026 001402                      BEQ      12$            ;NO, CONTINUE
6914 043030 000137 043524                      JMP      63$            ;YES, LOOP ON ERROR
6915
6916 043034 052737 002400 003524 12$:      BIS      #MCLK!MEWD,E.MR1 ;CREATE EXPECTED MAINT REG 1
6917 043042 012762 000440 000026      MOV      #DMD,MCLK,RKMR1(R2) ;PROVIDE 2ND UPWARD TRANSITION
6918 043050 016237 000026 003464      MOV      RKMR1(R2),T.MR1 ;STORE MAINT REG. 1
6919 043056 023737 003524 003464      CMP      E.MR1,T.MR1    ;CHECK IF MAINT REG. 1 CORRECT
6920 043064 001416                      BEQ      15$            ;YES, CONTINUE
6921 043066 012737 043106 001202      MOV      #13$,$ESCAPE   ;LOAD ESCAPE FOR LOOP ON ERROR
6922 043074 012737 064213 003172      MOV      #EMW3,EMW+2    ;LOAD ERROR MESSAGE
6923 043102 011646                      MOV      (SP),-(SP)     ;SAVE RETURN
6924 043104 000207                      RTS      PC              ;MR1 INCORRECT
6925
6926 043106 032777 001000 136024 13$:      BIT      #SW9,@SWR      ;CHECK IF LOOP ON ERROR
6927 043114 001402                      BEQ      15$            ;NO, CONTINUE
6928 043116 000137 043524                      JMP      63$            ;YES, LOOP ON ERROR
6929
6930 043122 052737 002000 003524 15$:      BIS      #MEWD,E.MR1    ;RESET TRANSITION INDICATION

```

6931	043130	042737	000400	003524		BIC	#MCLK,E.MR1	
6932	043136	012762	000040	000026		MOV	#DMD,RKMR1(R2)	;SUPPLY LAST PART OF DATA
6933	043144	016237	000026	003464		MOV	RKMR1(R2),T.MR1	;STORE MR1
6934	043152	023737	003464	003524		CMP	T.MR1,E.MR1	;CHECK IF MR1 CORRECT
6935	043160	001414				BEQ	18\$	;YES, RETURN
6936	043162	012737	043202	001202		MOV	#17\$, \$ESCAPE	;LOAD ESCAPE FOR LOOP ON ERROR
6937	043170	012737	064302	003172		MOV	#EMW4,EMW+2	;LOAD ERROR MESSAGE
6938	043176	011646				MOV	(SP),-(SP)	;SAVE RETURN
6939	043200	000207				RTS	PC	;MR1 INCORRECT
6940								
6941	043202	032777	001000	135730	17\$:	BIT	#SW9,@SWR	;CHECK IF LOOP ON ERROR
6942	043210	001145				BNE	63\$	;YES, LOOP ON ERROR
6943	043212	005037	001202		18\$:	CLR	\$ESCAPE	;CLEAR ESCAPE
6944	043216	062716	000002			ADD	#2,(SP)	;ADJUST RETURN
6945	043222	000207				RTS	PC	;RETURN
6946								
6947	043224	005737	003612		20\$:	TST	P1.BIT	;CHECK IN NEXT BIT A ONE
6948	043230	001007				BNE	30\$	;YES, CHECK IF PRECOMP DELAY
6949	043232	005737	003616			TST	M1.BIT	;CHECK FOR PRECOMP ADVANCE
6950	043236	001415				BEQ	40\$	;NO, CLOCK IN DATA BIT
6951	043240	052737	004000	003524		BIS	#PCA,E.MR1	;SET PRECOMP. ADVANCE
6952	043246	000411				BR	40\$	;CHECK MR1
6953								
6954	043250	042737	000400	003524	30\$:	BIC	#MCLK,E.MR1	;RESET MAINT CLOCK IN EXPECTED MR1
6955	043256	005737	003616			TST	M1.BIT	;CHECK FOR PRECOMP DELAY
6956	043262	001003				BNE	40\$	;NO, CHECK MR1
6957	043264	052737	010000	003524		BIS	#PCD,E.MR1	;SET SET PRECOMP DELAY
6958	043272	012762	000040	000026	40\$:	MOV	#DMD,RKMR1(R2)	;CLOCK IN DATA BIT
6959	043300	016237	000026	003464		MOV	RKMR1(R2),T.MR1	;STORE MR1
6960	043306	023737	003464	003524		CMP	T.MR1,E.MR1	;CHECK MR1 CORRECT
6961	043314	001414				BEQ	42\$	;YES, CLOCK IN REST OF BIT
6962	043316	012737	043336	001202		MOV	#41\$, \$ESCAPE	;LOAD ESCAPE FOR LOOP ON ERROR
6963	043324	012737	064122	003172		MOV	#EMW2,EMW+2	;LOAD ERROR MESSAGE
6964	043332	011646				MOV	(SP),-(SP)	;SAVE RETURN
6965	043334	000207				RTS	PC	;MR1 INCORRECT
6966								
6967	043336	032777	001000	135574	41\$:	BIT	#SW9,@SWR	;CHECK IF LOOP ON ERROR
6968	043344	001067				BNE	63\$	;YES, LOOP ON ERROR
6969	043346	052737	000400	003524	42\$:	BIS	#MCLK,E.MR1	;CHREATE EXPECTED MAINT. REG. 1
6970	043354	012762	000440	000026		MOV	#DMD.MCLK,RKMR1(R2)	;PROVIDE 2ND UPWARD TRANSITION
6971	043362	016237	000026	003464		MOV	RKMR1(R2),T.MR1	;STORE MAINT REG 1
6972	043370	023737	003524	003464		CMP	E.MR1,T.MR1	;CHECK IF MAINT REG 1 CORRECT
6973	043376	001414				BEQ	45\$	;YES, CONTINUE
6974	043400	012737	043420	001202		MOV	#43\$, \$ESCAPE	;LOAD ESCAPE
6975	043406	012737	064213	003172		MOV	#EMW3,EMW+2	;LOAD ERROR MESSAGE
6976	043414	011646				MOV	(SP),-(SP)	;SAVE RETURN
6977	043416	000207				RTS	PC	;MR1 INCORRECT
6978								
6979	043420	032777	001000	135512	43\$:	BIT	#SW9,@SWR	;CHECK IF LOOP ON ERROR
6980	043426	001036				BNE	63\$	;YES, LOOP ON ERROR
6981	043430	042737	002400	003524	45\$:	BIC	#MEWD.MCLK,E.MR1	;SET TRANSITION
6982	043436	012762	000040	000026		MOV	#DMD,RKMR1(R2)	;CLOCK TRANSITION
6983	043444	016237	000026	003464		MOV	RKMR1(R2),T.MR1	;STORE MR1
6984	043452	023737	003464	003524		CMP	T.MR1,E.MR1	;CHECK IF MR1 CORRECT
6985	043460	001414				BEQ	50\$	;YES, RETURN
6986	043462	012737	043502	001202		MOV	#47\$, \$ESCAPE	;LOAD ESCAPE FOR LOOP ON ERROR

```
6987 043470 012737 064302 003172      MOV    #EMW4,EMW+2      ;LOAD ERROR MESSAGE
6988 043476 011646                MOV    (SP),-(SP)      ;SAVE RETURN
6989 043500 000207                RTS    PC               ;MR1 INCORRECT
6990
6991 043502 032777 001000 135430 47$:  BIT    #SW9,@SWR      ;CHECK IF LOOP ON ERROR
6992 043510 001005                BNE   63$              ;YES, LOOP ON ERROR
6993 043512 005037 001202 50$:  CLR    $ESCAPE        ;CLEAR ESCAPE
6994 043516 062716 000002      ADD    #2,(SP)         ;ADJUST RETURN
6995 043522 000207                RTS    PC               ;RETURN
6996
6997 043524 005037 001202 63$:  CLR    $ESCAPE        ;CLEAR ESCAPE
6998 043530 012706 001100      MOV    #STACK,SP      ;FORCE STACK
6999 043534 000177 135350      JMP    @SLPERR         ;LOOP ON ERROR
7000
7001      .SBTTL  SIMULATE ONE BIT OR READ DATA IN MAINTENANCE MODE
7002
7003 043540 005737 003614  RDBIT: TST    PR.BIT          ;CHECK IF ONE
7004 043544 001024                BNE   10$              ;YES, SIMULATE ONE
7005 043546 005737 003616      TST    M1.BIT         ;CHECK IF PREVIOUS ONE
7006 043552 001404                BEQ   4$               ;NO, INSERT TRANSITION
7007 043554 012762 000440 000026  MOV    #DMD.MCLK,RKMR1(R2) ;YES, DO NOT INSERT TRANSITION
7008 043562 000403                BR    5$               ;CLOCK IN ZERO
7009
7010 043564 012762 001440 000026 4$:  MOV    #DMD!MCLK!MERD,RKMR1(R2) ;INSERT TRANSITION
7011 043572 012762 000040 000026 5$:  MOV    #DMD,RKMR1(R2) ;CLOCK IN ZERO
7012 043600 012762 000440 000026      MOV    #DMD!MCLK,RKMR1(R2)
7013 043606 012762 000040 000026      MOV    #DMD,RKMR1(R2)
7014 043614 000207                RTS    PC               ;RETURN
7015
7016 043616 012762 000440 000026 0$:  MOV    #DMD!MCLK,RKMR1(R2) ;CLOCK IN ONE
7017 043624 012762 000040 000026      MOV    #DMD,RKMR1(R2)
7018 043632 012762 001440 000026      MOV    #DMD!MCLK!MERD,RKMR1(R2)
7019 043640 012762 000040 000026      MOV    #DMD,RKMR1(R2)
7020 043646 000207                RTS    PC               ;RETURN
7021
7022      .SBTTL  MEMORY CHECK ENABLE TRAP
7023
7024 043650 012737 043664 001202 MEMERR: MOV    #10$, $ESCAPE    ;LOAD ESCAPE
7025 043656 011637 003604      MOV    (SP),TRAPPC    ;STORE PC
7026 043662 104147                ERROR 147              ;REPORT MEM PARITY ERROR
7027 043664 005037 001202 10$:  CLR    $ESCAPE        ;CLEAR ESCAPE
7028 043670 032777 001000 135242  BIT    #SW9,@SWR      ;CHECK IF LOOP ON ERROR
7029 043676 001001                BNE   15$              ;YES, FORCE STACK AND TRY AGAIN
7030 043700 000002                RTI                     ;NO, RETURN
7031
7032 043702 012706 001100 15$:  MOV    #STACK,SP      ;INITIALIZE STACK
7033 043706 000177 135176      JMP    @SLPERR         ;LOOP ON ERROR
7034
7035      .SBTTL  ROUTINE TO SIZE MEMORY
7036
7037      ;*****
7038      ;*CALL:
7039      ;*      JSR    PC,$SIZE
7040      ;*      RETURN
7041      ;*$LSTAD WILL CONTAIN:
7042      ;*      WITH K11 OPTION
```

-- LAST VIRTUAL ADDRESS OF THE LAST BANK

```

7043      ;*      WITHOUT KT11 OPTION      -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
7044      ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
7045      ;*$KT11 IS THE MEMORY MANAGEMENT KEY
7046      ;*$BIT07 = 0 DON'T USE MEMORY MANAGEMENT
7047      ;*      MUST BE SETUP BEFORE THE CALL
7048      ;*$BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
7049      ;*
7050      ;*      DETERMINED BY ROUTINE
7051      $SIZE:  MOV      R0,-(SP)      ;;SAVE R0 ON THE STACK
7052      MOV      R1,-(SP)      ;;SAVE R1 ON THE STACK
7053      MOV      R2,-(SP)      ;;SAVE R2 ON THE STACK
7054      MOV      R3,-(SP)      ;;SAVE R3 ON THE STACK
7055      MOV      @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
7056      MOV      @#ERRVEC+2,-(SP)
7057      MOV      SP,R0          ;;SAVE THE STACK POINTER
7058      ;;SET THE ERRVEC PS TO THE PRESENT PS
7059      TRAP
7060      MOV      (SP)+,@#ERRVEC+2  ;;PUSH OLD PSW AND PC ON STACK
7061      MOV      #3776,R1      ;;SAVE THE PSW IN @#ERRVEC+2
7062      TSTB     (PC)+          ;;SETUP ADDRESS
7063      $KT11:  .WORD    200      ;;USE MEMORY MANAGEMENT?
7064      BPL      $SCORE        ;;SET TO USE MEMORY MANAGEMENT
7065      MOV      #0,R1          ;;BR IF NO
7066      MOV      #0,R1          ;;SET FOR TIMEOUT
7067      TST      @#SR0         ;;KT11 ARE YOU THERE?
7068      BIS      #100000,$KT11  ;;YES--SET KT11 KEY
7069      CLR      -(SP)         ;;INITIALIZE FOR 'PAR' LOADING
7070      MOV      #KIPAR0,R2     ;;ADDRESS OF FIRST 'PAR'
7071      MOV      #^D8,R3       ;;LOAD EIGHT 'PAR.'S' AND EIGHT 'PDR.'S'
7072      MOV      #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
7073      MOV      (SP),(R2)+     ;;LOAD 'PAR'
7074      ADD      #200,(SP)      ;;UPDATE FOR NEXT 'PAR'
7075      SOB      R3,1$         ;;LOOP UNTIL ALL EIGHT ARE LOADED
7076      MOV      #177600,-(R2)  ;;SETUP KIPAR7 FOR I/O
7077      CLR      -(R2)         ;;SETUP KIPAR6 FOR TESTING
7078      MOV      #2$,@#ERRVEC   ;;CATCH TIMEOUT IF NO SR3
7079      MOV      #20,@#SR3     ;;ENABLE 22 BIT MODE
7080      BR       3$           ;;THIS PDP-11 HAS A SR3 REGISTER
7081      CMP      (SP)+,(SP)+    ;;CLEAN OFF THE STACK--NO SR3
7082      INC      @#SR0         ;;TURN ON MEMORY MANAGEMENT
7083      MOV      #0,R1          ;;SET FOR TIME OUT
7084      MOV      #0,R1          ;;TURN ON MEMORY MANAGEMENT
7085      MOV      #0,R1          ;;SET FOR TIME OUT
7086      TST      @#143776      ;;TRAP ON NON-EX-MEM
7087      ADD      #40,(R2)      ;;MAKE A 1K STEP
7088      CMP      @#KIPAR7,(R2)  ;;LAST ONE?
7089      BHI      4$           ;;NO--TRY IT
7090      MOV      (R2),R2       ;;GET LAST BANK+1
7091      CLR      @#SR0         ;;TURN OFF MEMORY MANAGEMENT
7092      BR       $SIZE        ;;TURN OFF MEMORY MANAGEMENT
7093      $KT11:  BIC      #100000,$KT11  ;;KT11 NON-EXISTENT
7094      MOV      #0,R1          ;;SET FOR TIMEOUT
7095      MOV      #0,R1          ;;SET FOR TIMEOUT
7096      CLR      R2           ;;SET UP BANK
7097      ADD      #4000,R1       ;;INCREMENT BY 1K
7098      ADD      #40,R2        ;;1K STEP
7099      TST      (R1)         ;;TRAP ON TIME OUT
7100      CMP      #177776,R1    ;;LAST ONE
7101      BNE      1$          ;;NO--TRY AGAIN
7102      SUB      #4000,R1

```

```
7099 044154 162702 000040 $SIZE: SUB #40,R2 ;;DROP BACK
7100 044160 010006 MOV R0,SP ;;RESTORE THE STACK
7101 044162 012637 000006 MOV (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
7102 044166 012637 000004 MOV (SP)+,@#ERRVEC
7103 044172 010137 044214 MOV R1,$LSTAD ;;LAST ADDRESS
7104 044176 010237 044216 MOV R2,$LSTBK ;;LAST BANK
7105 044202 012603 MOV (SP)+,R3 ;;RESTORE R3
7106 044204 012602 MOV (SP)+,R2 ;;RESTORE R2
7107 044206 012601 MOV (SP)+,R1 ;;RESTORE R1
7108 044210 012600 MOV (SP)+,R0 ;;RESTORE R0
7109 044212 000207 RTS PC
7110 044214 000000 $LSTAD: .WORD 0 ;;CONTAINS THE LAST ADDRESS
7111 044216 000000 $LSTBK: .WORD 0 ;;CONTAINS THE LAST BANK
7112 .SBTTL SCOPE HANDLER ROUTINE
7113
7114
7115 ::*****
7116 :*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
7117 :*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
7118 :*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
7119 :*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7120 :*SW14=1 LOOP ON TEST
7121 :*SW11=1 INHIBIT ITERATIONS
7122 :*SW09=1 LOOP ON ERROR
7123 :*SW08=1 LOOP ON TEST IN SWR<7:0>
7124 :*CALL
7125 :* SCOPE ;;SCOPE=IOT
7126 $SCOPE:
7127 044220 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
7128 044222 032777 040000 134710 $: BIT #BIT14,@SWR ;;LOOP ON PRESENT TEST?
7129 044230 001131 BNE $OVER ;;YES IF SW14=1
7130 :*****START OF CODE FOR THE XOR TESTER*****
7131 044232 000416 $XTSTR: BR 6$ ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
7132 MOV @#ERRVEC,-(SP) ;;THIS INSTRUCTION TO A 'NOP' (NOP-240)
7133 044234 013746 000004 MOV #5$,@#ERRVEC ;;SAVE THE CONTENTS OF THE ERROR VECTOR
7134 044240 012737 044260 000004 MOV #177060,($SP)+ ;;SET FOR TIMEOUT
7135 044246 005737 177060 TST @#177060 ;;TIME OUT ON XOR?
7136 044252 012637 000004 MOV (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
7137 044256 000500 BR $SVLAD ;;GO TO THE NEXT TEST
7138 044260 022626 5$: CMP (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
7139 044262 012637 000004 MOV (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
7140 044266 000440 BR 7$ ;;LOOP ON THE PRESENT TEST
7141 044270 6$:*****END OF CODE FOR THE XOR TESTER*****
7142 044270 032777 000400 134642 BIT #BIT08,@SWR ;;LOOP ON SPEC. TEST?
7143 044276 001421 BEQ 2$ ;;BR IF NO
7144 044300 005046 CLR -(SP) ;;CLEAR A TEMP. LOCATION
7145 044302 117716 134632 MOVB @SWR,(SP) ;;PICKUP THE DESIRED TEST NUMBER
7146 044306 001414 BEQ 8$ ;;BRANCH IF BAD TEST NUMBER IN SWR
7147 044310 022716 000051 CMP #51,(SP) ;;CHECK THE NUMBER IN THE SWR
7148 044314 002411 BLT 8$ ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
7149 044316 011637 001102 MOV (SP),$TSTNM ;;UPDATE THE TEST NUMBER
7150 044322 005316 DEC (SP) ;;BACKUP BY ONE
7151 044324 006316 ASL (SP) ;;SCALE THE TEST NUMBER AS AN INDEX
7152 044326 062716 044532 ADD #$$SW08TBL,(SP) ;;FORM THE ADDRESS OF TEST POINTER
7153 044332 013637 001106 MOV @(SP)+,$LPADR ;;SET LOOP ADDRESS TO DESIRED TEST
7154 044336 000466 BR $OVER ;;GO LOOP ON THE TEST
```



7155	044340	005726		8\$:	TST	(SP)+	::CLEAN THE BAD TEST NUMBER OFF OF THE STACK	
7156	044342	105737	001103	2\$:	TSTB	\$ERFLG	::HAS AN ERROR OCCURRED?	
7157	044346	001421			BEQ	3\$	::BR IF NO	
7158	044350	123737	001115	001103	CMPB	\$ERMAX,\$ERFLG	::MAX. ERRORS FOR THIS TEST OCCURRED?	
7159	044356	101015			BHI	3\$	::BR IF NO	
7160	044360	032777	001000	134552	BIT	#BIT09,@SWR	::LOOP ON ERROR?	
7161	044366	001404			BEQ	4\$	::BR IF NO	
7162	044370	013737	001110	001106	7\$:	MOV	\$LPERR,\$LPADR	::SET LOOP ADDRESS TO LAST SCOPE
7163	044376	000446			BR	\$OVER		
7164	044400	105037	001103		4\$:	CLRB	\$ERFLG	::ZERO THE ERROR FLAG
7165	044404	005037	001200			CLR	\$TIMES	::CLEAR THE NUMBER OF ITERATIONS TO MAKE
7166	044410	000415				BR	1\$	::ESCAPE TO THE NEXT TEST
7167	044412	032777	004000	134520	3\$:	BIT	#BIT11,@SWR	::INHIBIT ITERATIONS?
7168	044420	001011				BNE	1\$	::BR IF YES
7169	044422	005737	001222			TST	\$PASS	::IF FIRST PASS OF PROGRAM
7170	044426	001406				BEQ	1\$	::INHIBIT ITERATIONS
7171	044430	005237	001104			INC	\$ICNT	::INCREMENT ITERATION COUNT
7172	044434	023737	001200	001104		CMP	\$TIMES,\$ICNT	::CHECK THE NUMBER OF ITERATIONS MADE
7173	044442	002024				BGE	\$OVER	::BR IF MORE ITERATION REQUIRED
7174	044444	012737	000001	001104	1\$:	MOV	#1,\$ICNT	::REINITIALIZE THE ITERATION COUNTER
7175	044452	013737	044530	001200		MOV	\$MXCNT,\$TIMES	::SET NUMBER OF ITERATIONS TO DO
7176	044460	105237	001102		\$SVLAD:	INCB	\$STNM	::COUNT TEST NUMBERS
7177	044464	113737	001102	001220		MOVB	\$STNM,\$TESTN	::SET TEST NUMBER IN APT MAILBOX
7178	044472	011637	001106			MOV	(SP),\$LPADR	::SAVE SCOPE LOOP ADDRESS
7179	044476	011637	001110			MOV	(SP),\$LPERR	::SAVE ERROR LOOP ADDRESS
7180	044502	005037	001202			CLR	\$ESCAPE	::CLEAR THE ESCAPE FROM ERROR ADDRESS
7181	044506	112737	000001	001115		MOVB	#1,\$ERMAX	::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
7182	044514	013777	001102	134420	\$OVER:	MOV	\$STNM,@DISPLAY	::DISPLAY TEST NUMBER
7183	044522	013716	001106			MOV	\$LPADR,(SP)	::FUDGE RETURN ADDRESS
7184	044526	000002				RTI		::FIXES PS
7185	044530	003720			\$MXCNT:	2000.	::MAX. NUMBER OF ITERATIONS	
7186	044532				\$SW08TRI:			
7187	044532	004652			.WORD	TST1+2	::STARTING ADDRESS OF TEST 1	
7188	044534	005070			.WORD	TST2+2	::STARTING ADDRESS OF TEST 2	
7189	044536	005306			.WORD	TST3+2	::STARTING ADDRESS OF TEST 3	
7190	044540	005522			.WORD	TST4+2	::STARTING ADDRESS OF TEST 4	
7191	044542	005736			.WORD	TST5+2	::STARTING ADDRESS OF TEST 5	
7192	044544	006420			.WORD	TST6+2	::STARTING ADDRESS OF TEST 6	
7193	044546	007046			.WORD	TST7+2	::STARTING ADDRESS OF TEST 7	
7194	044550	007434			.WORD	TST10+2	::STARTING ADDRESS OF TEST 10	
7195	044552	010560			.WORD	TST11+2	::STARTING ADDRESS OF TEST 11	
7196	044554	011246			.WORD	TST12+2	::STARTING ADDRESS OF TEST 12	
7197	044556	012132			.WORD	TST13+2	::STARTING ADDRESS OF TEST 13	
7198	044560	013020			.WORD	TST14+2	::STARTING ADDRESS OF TEST 14	
7199	044562	014224			.WORD	TST15+2	::STARTING ADDRESS OF TEST 15	
7200	044564	015102			.WORD	TST16+2	::STARTING ADDRESS OF TEST 16	
7201	044566	015540			.WORD	TST17+2	::STARTING ADDRESS OF TEST 17	
7202	044570	016654			.WORD	TST20+2	::STARTING ADDRESS OF TEST 20	
7203	044572	017770			.WORD	TST21+2	::STARTING ADDRESS OF TEST 21	
7204	044574	021104			.WORD	TST22+2	::STARTING ADDRESS OF TEST 22	
7205	044576	021774			.WORD	TST23+2	::STARTING ADDRESS OF TEST 23	
7206	044600	022526			.WORD	TST24+2	::STARTING ADDRESS OF TEST 24	
7207	044602	023260			.WORD	TST25+2	::STARTING ADDRESS OF TEST 25	
7208	044604	024012			.WORD	TST26+2	::STARTING ADDRESS OF TEST 26	
7209	044606	024544			.WORD	TST27+2	::STARTING ADDRESS OF TEST 27	
7210	044610	025276			.WORD	TST30+2	::STARTING ADDRESS OF TEST 30	

```
7211 044612 026030 .WORD TST31+2 ;; STARTING ADDRESS OF TEST 31
7212 044614 026330 .WORD TST32+2 ;; STARTING ADDRESS OF TEST 32
7213 044616 027076 .WORD TST33+2 ;; STARTING ADDRESS OF TEST 33
7214 044620 027466 .WORD TST34+2 ;; STARTING ADDRESS OF TEST 34
7215 044622 030312 .WORD TST35+2 ;; STARTING ADDRESS OF TEST 35
7216 044624 031136 .WORD TST36+2 ;; STARTING ADDRESS OF TEST 36
7217 044626 031762 .WORD TST37+2 ;; STARTING ADDRESS OF TEST 37
7218 044630 032606 .WORD TST40+2 ;; STARTING ADDRESS OF TEST 40
7219 044632 033432 .WORD TST41+2 ;; STARTING ADDRESS OF TEST 41
7220 044634 034256 .WORD TST42+2 ;; STARTING ADDRESS OF TEST 42
7221 044636 035102 .WORD TST43+2 ;; STARTING ADDRESS OF TEST 43
7222 044640 035742 .WORD TST44+2 ;; STARTING ADDRESS OF TEST 44
7223 044642 036766 .WORD TST45+2 ;; STARTING ADDRESS OF TEST 45
7224 044644 040012 .WORD TST46+2 ;; STARTING ADDRESS OF TEST 46
7225 044646 040652 .WORD TST47+2 ;; STARTING ADDRESS OF TEST 47
7226 044650 041276 .WORD TST50+2 ;; STARTING ADDRESS OF TEST 50
7227 044652 041722 .WORD TST51+2 ;; STARTING ADDRESS OF TEST 51
7228 *****
7229 .SBTTL LOOP ON INTERNAL ERROR
7230
7231 044654 032777 001000 134256 SCOP1$: BIT #SW9,@SWR ;; CHECK IF LOOP ON ERROR
7232 044662 001405 BEQ 5$ ;; NO RETURN
7233 044664 105737 001103 TSTB $ERFLG ;; CHECK IF ERROR OCCURED
7234 044670 001402 BEQ 5$ ;; NO, RETURN
7235 044672 013716 001110 MOV $LPERR,(SP) ;; GO BACK TO BEGINNING OF LOOP
7236 044676 000002 5$: RTI ;; RETURN
7237 .SBTTL APT COMMUNICATIONS ROUTINE
7238 *****
7239
7240 044700 112737 000001 045144 $ATY1: MOVB #1,$FFLG ;; TO REPORT FATAL ERROR
7241 044706 112737 000001 045142 $ATY3: MOVB #1,$MFLG ;; TO TYPE A MESSAGE
7242 044714 000403 BR $ATYC
7243 044716 112737 000001 045144 $ATY4: MOVB #1,$FFLG ;; TO ONLY REPORT FATAL ERROR
7244 044724 $ATYC:
7245 044724 010046 MOV R0,-(SP) ;; PUSH R0 ON STACK
7246 044726 010146 MOV R1,-(SP) ;; PUSH R1 ON STACK
7247 044730 105737 045142 TSTB $MFLG ;; SHOULD TYPE A MESSAGE?
7248 044734 001450 BEQ 5$ ;; IF NOT: BR
7249 044736 122737 000001 001234 CMPB #APTENV,$ENV ;; OPERATING UNDER APT?
7250 044744 001031 BNE 3$ ;; IF NOT: BR
7251 044746 132737 000100 001235 BITB #APTSPOOL,$ENVM ;; SHOULD SPOOL MESSAGES?
7252 044754 001425 BEQ 3$ ;; IF NOT: BR
7253 044756 017600 000004 MOV @4(SP),R0 ;; GET MESSAGE ADDR.
7254 044762 062766 000002 000004 ADD #2,4(SP) ;; BUMP RETURN ADDR.
7255 044770 005737 001214 1$: TST $MSGTYPE ;; SEE IF DONE W/ LAST XMISSION?
7256 044774 001375 BNE 1$ ;; IF NOT: WAIT
7257 044776 010037 001230 MOV R0,$MSGAD ;; PUT ADDR IN MAILBOX
7258 045002 105720 2$: TSTB (R0)+ ;; FIND END OF MESSAGE
7259 045004 001376 BNE 2$
7260 045006 163700 001230 SJB $MSGAD,R0 ;; SUB START OF MESSAGE
7261 045012 006200 ASR R0 ;; GET MESSAGE LNTH IN WORDS
7262 045014 010037 001232 MOV R0,$MSGGLT ;; PUT LENGTH IN MAILBOX
7263 045020 012737 000004 001214 MOV #4,$MSGTYPE ;; TELL APT TO TAKE MSG.
7264 045026 000413 BR 5$
7265 045030 017637 000004 045054 3$: MOV @4(SP),4$ ;; PUT MSG ADDR IN JSR LINKAGE
7266 045036 062766 000002 000004 ADD #2,4(SP) ;; BUMP RETURN ADDRESS
```

```

7267 045044 013746 177776      MOV      177776,-(SP)      ;;PUSH 177776 ON STACK
7268 045050 004737 045706      JSR      PC,$TYPE        ;;CALL TYPE MACRO
7269 045054 000000      4$:      .WORD      0
7270 045056      5$:
7271 045056 105737 045144      10$:     TSTB      $FFLG        ;; SHOULD REPORT FATAL ERROR?
7272 045062 001416      BEQ      12$             ;; IF NOT: BR
7273 045064 005737 001234      TST      $ENV           ;; RUNNING UNDER APT?
7274 045070 001413      BEQ      12$           ;; IF NOT: BR
7275 045072 005737 001214      11$:     TST      $MSGTYPE      ;; FINISHED LAST MESSAGE?
7276 045076 001375      BNE      11$           ;; IF NOT: WAIT
7277 045100 017637 000004 001214     MOV      @4(SP),$FATAL   ;; GET ERROR #
7278 045106 062766 000002 000004     ADD      #2,4(SP)       ;; BUMP RETURN ADDR.
7279 045114 005237 001214      INC      $MSGTYPE       ;; TELL APT TO TAKE ERROR
7280 045120 105037 045144      12$:     CLRB      $FFLG        ;; CLEAR FATAL FLAG
7281 045124 105037 045143      CLRB      $LFLG        ;; CLEAR LOG FLAG
7282 045130 105037 045142      CLRB      $MFLG        ;; CLEAR MESSAGE FLAG
7283 045134 012601      MOV      (SP)+,R1       ;; POP STACK INTO R1
7284 045136 012600      MOV      (SP)+,R0       ;; POP STACK INTO R0
7285 045140 000207      RTS      PC             ;; RETURN
7286 045142      000      $MFLG: .BYTE 0   ;; MESSG. FLAG
7287 045143      000      $LFLG: .BYTE 0   ;; LOG FLAG
7288 045144      000      $FFLG: .BYTE 0   ;; FATAL FLAG
7289      045146      .EVEN
7290      000200      APTSIZE=200
7291      000001      APTENV=001
7292      000100      APTSPOOL=100
7293      000040      APTCSUP=040
7294      .SBTTL  ERROR HANDLER ROUTINE
7295
7296      ;;*****
7297      ;;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
7298      ;;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
7299      ;;*AND GO TO TYPERR ON ERROR
7300      ;;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7301      ;;*SW15=1      HALT ON ERROR
7302      ;;*SW13=1      INHIBIT ERROR TYPEOUTS
7303      ;;*SW10=1      BELL ON ERROR
7304      ;;*SW09=1      LOOP ON ERROR
7305      ;;*CALL
7306      ;;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
7307
7308      $ERROR:
7309 045146 104407      7$:      CKSWR      $ERFLG        ;; TEST FOR CHANGE IN SOFT-SWR
7310 045150 105237 001103      INCB      $ERFLG        ;; SET THE ERROR FLAG
7311 045154 001775      BEQ      7$            ;; DON'T LET THE FLAG GO TO ZERO
7312 045156 013777 001102 133756     MOV      $TSTNM,@DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
7313 045164 032777 002000 133746     BIT      #BIT10,@SWR    ;; BELL ON ERROR?
7314 045172 001402      BEQ      1$           ;; NO - SKIP
7315 045174 104401 001204      TYPE      $BELL        ;; RING BELL
7316 045200 005237 001112      1$:      INC      $ERTIL       ;; COUNT THE NUMBER OF ERRORS
7317 045204 011637 001116      MOV      (SP),$ERRPC    ;; GET ADDRESS OF ERROR INSTRUCTION
7318 045210 162737 000002 001116     SUB      #2,$ERRPC
7319 045216 117737 133674 001114     MOV      @$ERRPC,$ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
7320 045224 032777 020000 133706     BIT      #BIT13,@SWR    ;; SKIP TYPEOUT IF SET
7321 045232 001004      BNE      20$          ;; SKIP TYPEOUTS
7322 045234 004737 045346      JSR      PC,TYPERR     ;; GO TO USER ERROR ROUTINE
  
```

```

7323 045240 10440 001211
7324 045244
7325 045244 122737 000001 001234
7326 045252 001007
7327 045254 113737 001114 045266
7328 045262 004737 044716
7329 045266 000
7330 045267 000
7331 045270 000777
7332 045272 005777 133642
7333 045276 100002
7334 045300 000000
7335 045302 104407
7336 045304 032777 001000 133626
7337 045312 001402
7338 045314 013716 001110
7339 045320 005737 001202
7340 045324 001402
7341 045326 013716 001202
7342 045332
7343 045332 022737 042550 000042
7344 045340 001001
7345 045342 000000
7346 045344
7347 045344 000002
7348
7349
7350
7351
7352
7353
7354
7355
7356
7357
7358
7359 045346 104413
7360 045350 113700 001114
7361 045354 042700 177400
7362 045360 005300
7363 045362 006300
7364 045364 006300
7365 045366 006300
7366 045370 062700 001300
7367 045374 012037 045410
7368 045400 001404
7369 045402 104401 001211
7370 045406 104401
7371 045410 000000
7372 045412 012037 045426
7373 045416 001404
7374 045420 104401 001211
7375 045424 104401
7376 045426 000000
7377 045430 012001
7378 045432 001445

      TYPE      ,SCLF
20$:  CMPB      #APTENV,$ENV      ;;RUNNING IN APT MODE
      BNE       2$                ;;NO,SKIP APT ERROR REPORT
      MOVB      $ITEMB,21$        ;;SET ITEM NUMBER AS ERROR NUMBER
      JSR       PC,$ATY4          ;;REPORT FATAL ERROR TO APT
21$:  .BYTE     0
      .BYTE     0
22$:  BR        22$                ;;APT ERROR LOOP
2$:   TST       @SWR              ;;HALT ON ERROR
      BPL       3$                ;;SKIP IF CONTINUE
      HALT      ;;HALT ON ERROR!
3$:   CKSWR
      BIT       #BIT09,@SWR      ;;TEST FOR CHANGE IN SOFT-SWR
      BEQ       4$                ;;LOOP ON ERROR SWITCH SET?
4$:   MOV       $LPERR,(SP)       ;;FUDGE RETURN FOR LOOPING
      TST       $ESCAPE          ;;CHECK FOR AN ESCAPE ADDRESS
      BEQ       5$                ;;BR IF NONE
      MOV       $ESCAPE,(SP)     ;;FUDGE RETURN ADDRESS FOR ESCAPE
5$:   CMP       #SENDAD,@#42      ;;ACT-11 AUTO-ACCEPT?
      BNE       6$                ;;BRANCH IF NO
      HALT
6$:   RTI                          ;;RETURN

;*****
;SBTTL TYPE ERROR ROUTINE
;*ENTRY JSR PC,TYPERR
;*RETURN RTS PC
;
;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRIB)
;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
;*THE ERROR.
;*****
TYPERR: SAVREG
      MOVB      $ITEMB,R0          ;ENTER ERROR NUMBER
      BIC       #177400,R0        ;CLEAR UNUSED BITS
      DEC       R0                 ;FORM INDEX FOR ERROR TABLE
      ASL       R0
      ASL       R0
      ASL       R0
1$:   ADD       #SERRTB,R0         ;FORM ADDRESS OF ERROR ENTRY
      MOV       (R0)+,2$          ;GET EM POINTER
      BEQ       3$                ;BRANCH IF THERE ISN'T ONE
      TYPE      ,SCLF             ;TYPE CARRIAGE RETURN LINE FEED
      TYPE      ;TYPE ERROR MESSAGE (EM)
2$:   .WORD     0                  ;EM POINTER GOES HERE
3$:   MOV       (R0)+,4$          ;GET DH POINTER
      BEQ       5$                ;BRANCH IF THERE ISN'T ONE
      TYPE      ,SCLF             ;TYPE CR-LF
      TYPE      ;TYPE DATA HEADER
4$:   .WORD     0                  ;DH POINTER GOES HERE
5$:   MOV       (R0)+,R1          ;GET DT POINTER
      BEQ       20$              ;BRANCH IF THERE ARE NONE

```

7379	045434	005004		CLR	R4		:RESET INDENT SWITCH
7380	045436	012000		MOV	(R0)+,R0		:GET DF POINTER
7381	045440	012002		MOV	(R0)+,R2		:STORE NUMBER OF DH'S
7382	045442	104401	001211	TYPE	,SCLF		
7383	045446	112003		10\$:	MOVB	(R0)+,R3	:GET & STORE NUMBER OF DATA WORDS
7384	045450	105720			TSTB	(R0)+	:BUMP PAST FORMAT WORD
7385	045452	005703			TST	R3	:TEST IF ANY DATA FOR THIS HEADER
7386	045454	001416			BEQ	14\$	:NO - SKIP DATA PRINT
7387	045456	005704			TST	R4	:CHECK IF INDENT WORDS
7388	045460	001004			BNE	12\$	:YES, GO INDENT
7389	045462	013146		11\$:	MOV	@(R1)+,-(SP)	:PUT FIRST DATA WORD ON STACK
7390	045464	104402			TYPOC		:TYPE IT
7391	045466	005303			DEC	R3	:MORE DATA WORDS
7392	045470	001403			BEQ	13\$	:NO-BRANCH
7393	045472	104401	052353	12\$:	TYPE	,SPACE2	:TYPE SEPARATORS
7394	045476	000771			BR	11\$	:LOOP
7395	045500	104401	001211	13\$:	TYPE	,SCLF	:TYPE <CR><LF>
7396	045504	005710			TST	(R0)	:CHECK IF NEXT HEADER AVAILABLE
7397	045506	001401			BEQ	14\$	:NO, DO NOT CHANGE INDENT
7398	045510	005104			COM	R4	:CHANGE INDENT
7399	045512	005302		14\$:	DFC	R2	:MORE DH'S?
7400	045514	003414			BLE	20\$	:NO-BRANCH
7401	045516	012037	045536	15\$:	MOV	(R0)+,18\$	:GET NEXT DH POINTER
7402	045522	001751			BEQ	10\$	:IF NO HEADER GET DATA
7403	045524	005704			TST	R4	:INDENT?
7404	045526	001402			BEQ	17\$	:NO-BRANCH
7405	045530	104401	052353		TYPE	,SPACE2	:INDENT
7406	045534	104401		17\$:	TYPE		:TYPE DH
7407	045536	000000		12\$:	.WORD	0	:DH POINTER GOES HERE
7408	045540	104401	001211		TYPE	,SCLF	
7409	045544	000740			BR	10\$	:LOOP
7410	045546	104414		20\$:	RESREG		
7411	045550	005237	003610		INC	ERRCNT	:INCREMENT ERROR COUNT
7412	045554	032777	010000	133356	BIT	#SW12,@SWR	:CHECK IF ABORT AFTER 20 ERRORS
7413	045562	001421			BEQ	25\$	:NO, RETURN
7414	045564	022737	000024	003610	CMP	#20,ERRCNT	:CHECK IF ERROR THRESHOLD EXCEEDED
7415	045572	103015			BHS	25\$	:NO, RETURN
7416	045574	104401	052356		TYPE	,ABORT	:TYPE 'PROGRAM HAS BEEN ABORTED BECAUSE
7417							: ERROR THRESHOLD EXCEEDED'
7418	045600	005737	000042		TST	42	:CHECK IF IN CHAIN MODE
7419	045604	001407			BEQ	30\$	:NO, HALT
7420	045606	012737	000001	042406	MOV	#1,\$EOPCT	:FORCE END OF PASS COUNT TO ON FOR ABORT
7421	045614	012706	001100		MOV	#STACK,SP	:INITIALIZE STACK
7422	045620	000137	042360		JMP	\$EOP	:BRING IN NEXT PROGRAM IN CHAIN
7423	045624	000000		30\$:	HALT		
7424	045626	000207		25\$:	RTS	PC	
7425							
7426				.SBTTL	CONTROLLED PROGRAM HALT		
7427	045630	013702	001270		CTRHL*: MOV	\$BASE,R2	:SET '611 BASE
7428	045634	012762	100000	000000	MOV	#CLR,RKCS1(R2)	:CLEAR CONTROLLER
7429	045642	012706	001100		MOV	#STACK,SP	:CLEAR STACK
7430	045646	104401	052307		TYPE	,OPROOF	:TYPE HALT MESSAGE
7431	045652	005737	000042		TST	42	:TEST IF MONITOR PRESENT
7432	045656	001410			BEQ	5\$	:NO - SKIP
7433	045660	105037	001103		CLRB	\$ERFLG	:CLEAR ERROR FLAG
7434	045664	005037	001202		CLR	\$ESCAPE	:CLEAR ESCAPE

```

7435 045670 005037 042406 CLR $EOPCT ;SET PASS COUNT TO 0
7436 045674 000137 042360 JMP $EOP ;GO TO END OF PASS
7437
7438 045700 000000 5$: HALT ;HALT PROGRAM
7439 045702 000137 003666 JMP $START1 ;DO RESTART IF CONTINUE
7440
7441 ;SRTT TYPE ROUTINE
7442
7443
7444 ;*****
7445 ;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
7446 ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
7447 ;NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
7448 ;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
7449 ;NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
7450
7451 ;CALL:
7452 ;*1) USING A TRAP INSTRUCTION ;:MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
7453 ;* TYPE ,MESADR
7454 ;*OR
7455 ;* TYPE
7456 ;* MFADR
7457 ;*
7458 045706 105737 001157 $TYPE: TSTB $TFPLG ;:IS THERE A TERMINAL?
7459 045712 100002 BPL 1$ ;:BR IF YES
7460 045714 000000 HALT ;:HALT HERE IF NO TERMINAL
7461 045716 000430 BR 3$ ;:LEAVE
7462 045720 010046 1$: MOV RO,-(SP) ;:SAVE RO
7463 045722 017600 000002 MOV @2(SP),RO ;:GET ADDRESS OF ASCIZ STRING
7464 045726 122737 000001 001234 (MPB #APTENV,$ENV ;:RUNNING IN APT MODE
7465 045734 001011 BNE 62$ ;:NO,GO CHECK FOR APT CONSOLE
7466 045736 132737 000100 001235 BITB #APTPOOL,$ENVM ;:SPOOL MESSAGE TO APT
7467 045744 001405 BEQ 62$ ;:NO,GO CHECK FOR CONSOLE
7468 045746 010037 045756 MOV RO,61$ ;:SETUP MESSAGE ADDRESS FOR APT
7469 045752 004737 044706 JSR PC,$ATY3 ;:SPOOL MESSAGE TO APT
7470 045756 000000 61$: .WORD 0 ;:MESSAGE ADDRESS
7471 045760 132737 000040 ( 1235 62$: BITB #APTCSUP,$ENVM ;:APT CONSOLE SUPPRESSED
7472 045766 001003 BNE 60$ ;:YES,SKIP TYPE OUT
7473 045770 112046 2$: MOVB (RO)+,-(SP) ;:PUSH CHARACTER TO BE TYPED ONTO STACK
7474 045772 001005 BNE 4$ ;:BR IF IT ISN'T THE TERMINATOR
7475 045774 005726 TST (SP)+ ;:IF TERMINATOR POP IT OFF THE STACK
7476 045776 012600 60$: MOV (SP)+,RO ;:RESTORE RO
7477 046000 062716 000002 3$: ADD #2,(SP) ;:ADJUST RETURN PC
7478 046004 000002 RTI ;:RETURN
7479 046006 122716 000011 4$: (MPB #HT,(SP) ;:BRANCH IF <HT>
7480 046012 001430 BEQ 8$
7481 046014 122716 000200 (MPB #CRLF,(SP) ;:BRANCH IF NOT <CRLF>
7482 046020 001006 BNE 5$
7483 046022 005726 TST (SP)+ ;:POP <CR><LF> EQUIV
7484 046024 104401 TYPE ;:TYPE A CR AND LF
7485 046026 001211 $CRLF
7486 046030 105037 046236 CLRB $CHARCNT ;:CLEAR CHARACTER COUNT
7487 046034 000755 BR 2$ ;:GET NEXT CHARACTER
7488 046036 004737 (46120 5$: JSR PC,$TYPEC ;:GO TYPE THIS CHARACTER
7489 046042 123726 001156 6$: (MPB $FILLC,(SP)+ ;:IS IT TIME FOR FILLER CHARS.?
7490 046046 001350 BNE 2$ ;:IF NO GO GET NEXT CHAR.

```

```

7491 046050 013746 001154          MOV    $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
7492                                     ;;AND THE NULL CHAR.
7493 046054 105366 000001      7$:   DECB    1(SP)      ;;DOES A NULL NEED TO BE TYPED?
7494 046060 002770                                     ;;BR IF NO--GO POP THE NULL OFF OF STACK
7495 046062 004737 046120          JSR    PC,$TYPEC      ;;GO TYPE A NULL
7496 046066 105337 046236          DECB    $CHARCNT      ;;DO NOT COUNT AS A COUNT
7497 046072 000770          BR     7$            ;;LOOP
7498
7499                                     ;HORIZONTAL TAB PROCESSOR
7500
7501 046074 112716 000040      8$:   MOVB   #' ,(SP)      ;;REPLACE TAB WITH SPACE
7502 046100 004737 046120      9$:   JSR    PC,$TYPEC      ;;TYPE A SPACE
7503 046104 132737 000007      046236 BITB   #7,$CHARCNT      ;;BRANCH IF NOT AT
7504 046112 001372                                     ;;TAB STOP
7505 046114 005726          TST    (SP)+          ;;POP SPACE OFF STACK
7506 046116 000724          BR     2$            ;;GET NEXT CHARACTER
7507 046120          $TYPEC:
7508 046120 105777 133020          TSTB   @STKS          ;;CHAR IN KYBD BUFFER?
7509 046124 100022          BPL    10$           ;;BR IF NOT
7510 046126 017746 133014          MOV    @STKB,-(SP)      ;;GET CHAR
7511 046132 042716 177600          BIC   #177600,(SP)      ;;STRIP EXTRANEIOUS BITS
7512 046136 122716 000023          CMPB   #$XOFF,(SP)      ;;WAS CHAR XOFF
7513 046142 001012          BNE    102$          ;;BR IF NOT
7514 046144
7515 046144 105777 132774      101$: TSTB   @STKS          ;;WAIT FOR CHAR
7516 046150 100375          BPL    101$          ;;BR IF NOT
7517 046152 117716 132770          MOVB   @STKB,(SP)      ;;GET CHAR
7518 046156 042716 177600          BIC   #177600,(SP)      ;;STRIP IT
7519 046162 122716 000021          CMPB   #$XON,(SP)      ;;WAS IT XON?
7520 046166 001366          BNE    101$          ;;BR IF NOT
7521 046170
7522 046170 005726      102$: TST    (SP)+          ;;FIX STACK
7523 046172
7524 046172 105777 132752      10$: TSTB   @STPS          ;;WAIT UNTIL PRINTER IS READY
7525 046176 100375          BPL    10$           ;;BR IF NOT
7526 046200 116677 000002      132744 MOVB   2(SP),@STPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
7527 046206 122766 000015      000002 CMPB   #CR,2(SP)        ;;IS CHARACTER A CARRIAGE RETURN?
7528 046214 001003          BNE    1$            ;;BRANCH IF NO
7529 046216 105037 046236          CLRB   $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
7530 046222 000406          BR     $TYPEX        ;;EXIT
7531 046224 122766 000012      000002 1$:   CMPB   #LF,2(SP)      ;;IS CHARACTER A LINE FEED?
7532 046232 001402          BEQ    $TYPEX        ;;BRANCH IF YES
7533 046234 105227          INCB   (PC)+          ;;COUNT THE CHARACTER
7534 046236 000000          $CHARCNT: .WORD 0      ;;CHARACTER COUNT STORAGE
7535 046240 000207          $TYPEX: RTS    PC
7536
7537                                     .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
7538
7539                                     ;*****
7540                                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
7541                                     ;*OCTAL (ASCII) NUMBER AND TYPE IT.
7542                                     ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
7543                                     ;*CALL:
7544                                     ;*   MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
7545                                     ;*   TYPOS      ;;CALL FOR TYPEOUT
7546                                     ;*   .BYTE  N            ;;N-1 TO 6 FOR NUMBER OF DIGITS TO TYPE

```

```

7547          .BYTE      M          ;;M=1 OR 0
7548          ;;1=TYPE LEADING ZEROS
7549          ;;0=SUPPRESS LEADING ZEROS
7550
7551          *$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
7552          *$TYPOS OR $TYPOC
7553          *CALL:
7554          *      MOV      NJM,-(SP)      ;;NUMBER TO BE TYPED
7555          *      TYPON                      ;;CALL FOR TYPEOUT
7556
7557          *$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
7558          *CALL:
7559          *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7560          *      TYPOC                      ;;CALL FOR TYPEOUT
7561
7562 046242 017646 000000          $TYPOS: MOV      @(SP),-(SP)      ;;PICKUP THE MODE
7563 046246 116637 000001 046465  MOVB     1(SP), $OFILL      ;;LOAD ZERO FILL SWITCH
7564 046254 112637 046467          MOVB     (SP)+, $OMODE+1    ;;NUMBER OF DIGITS TO TYPE
7565 046260 062716 000002          ADD      #2,(SP)          ;;ADJUST RETURN ADDRESS
7566 046264 000406          BR      $TYPON
7567 046266 112737 000001 046465  $TYPOC: MOVB     #1, $OFILL      ;;SET THE ZERO FILL SWITCH
7568 046274 112737 000006 046467  MOVB     #6, $OMODE+1    ;;SET FOR SIX(6) DIGITS
7569 046302 112737 000005 046464  $TYPON: MOVB     #5, $OCNT      ;;SET THE ITERATION COUNT
7570 046310 010346          MOV      R3,-(SP)        ;;SAVE R3
7571 046312 010446          MOV      R4,-(SP)        ;;SAVE R4
7572 046314 010546          MOV      R5,-(SP)        ;;SAVE R5
7573 046316 113704 046467          MOVB     $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
7574 046322 005404          NEG      R4
7575 046324 062704 000006          ADD      #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
7576 046330 110437 046466          MOVB     R4, $OMODE      ;;SAVE IT FOR USE
7577 046334 113704 046465          MOVB     $OFILL,R4      ;;GET THE ZERO FILL SWITCH
7578 046340 006605 000012          MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
7579 046344 005003          CLR      R3             ;;CLEAR THE OUTPUT WORD
7580 046346 006105          1$:  ROL      R5           ;;ROTATE MSB INTO 'C'
7581 046350 000404          BR      3$             ;;GO DO MSB
7582 046352 006105          2$:  ROL      R5           ;;FORM THIS DIGIT
7583 046354 006105          ROL      R5
7584 046356 006105          ROL      R5
7585 046360 010503          MOV      R5,R3
7586 046362 006103          3$:  ROL      R3           ;;GET LSB OF THIS DIGIT
7587 046364 105337 046466          DECB     $OMODE         ;;TYPE THIS DIGIT?
7588 046370 100016          BPL      7$            ;;BR IF NO
7589 046372 042703 177770          BIC      #177770,R3     ;;GET RID OF JUNK
7590 046376 001002          BNE      4$            ;;TEST FOR 0
7591 046400 005704          TST      R4             ;;SUPPRESS THIS 0?
7592 046402 001403          BEQ      5$            ;;BR IF YES
7593 046404 005204          4$:  INC      R4           ;;DON'T SUPPRESS ANYMORE 0'S
7594 046406 052703 000060          BIS      #'0,R3        ;;MAKE THIS DIGIT ASCII
7595 046412 052703 000040          5$:  BIS      #' ,R3      ;;MAKE ASCII IF NOT ALREADY
7596 046416 110337 046462          MOVB     R3,8$         ;;SAVE FOR TYPING
7597 046422 104401 046462          TYPE     ,8$          ;;GO TYPE THIS DIGIT
7598 046426 105337 046464          7$:  DECB     $OCNT      ;;COUNT BY 1
7599 046432 003347          BGT      2$            ;;BR IF MORE TO DO
7600 046434 002402          BLT      6$            ;;BR IF DONE
7601 046436 005204          INC      R4             ;;INSURE LAST DIGIT ISN'T A BLANK
7602 046440 000744          BR      2$            ;;GO DO THE LAST DIGIT

```



```

7603 046442 012605      6$:  MOV    (SP)+,R5      ;;RESTORE R5
7604 046444 012604      MOV    (SP)+,R4      ;;RESTORE R4
7605 046446 012603      MOV    (SP)+,R3      ;;RESTORE R3
7606 046450 016666 000002 000004  MOV    2(SP),4(SP)   ;;SET THE STACK FOR RETURNING
7607 046456 012616      MOV    (SP)+,(SP)
7608 046460 000002      RTI                      ;;RETURN
7609 046462      000      8$:  .BYTE  0              ;;STORAGE FOR ASCII DIGIT
7610 046463      000      .BYTE  0              ;;TERMINATOR FOR TYPE ROUTINE
7611 046464      000      $OCNT: .BYTE  0          ;;OCTAL DIGIT COUNTER
7612 046465      000      $OFILL: .BYTE  0        ;;ZERO FILL SWITCH
7613 046466 000000      $UMODE: .WORD  0       ;;NUMBER OF DIGITS TO TYPE
7614      .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
7615
7616      ;:*****
7617      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
7618      ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
7619      ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
7620      ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
7621      ;*REPLACED WITH SPACES.
7622      ;*CALL:
7623      ;*  MOV    NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
7624      ;*  TYPDS                      ;;GO TO THE ROUTINE
7625
7626      $TYPDS:
7627 046470 010046      MOV    R0,-(SP)      ;;PUSH R0 ON STACK
7628 046472 010146      MOV    R1,-(SP)      ;;PUSH R1 ON STACK
7629 046474 010246      MOV    R2,-(SP)      ;;PUSH R2 ON STACK
7630 046476 010346      MOV    R3,-(SP)      ;;PUSH R3 ON STACK
7631 046500 010546      MOV    R5,-(SP)      ;;PUSH R5 ON STACK
7632 046502 012746 020200      MOV    #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
7633 046506 016605 000020      MOV    20(SP),R5     ;;GET THE INPUT NUMBER
7634 046512 100004      BPL    1$            ;;BR IF INPUT IS POS.
7635 046514 005405      NEG    R5            ;;MAKE THE BINARY NUMBER POS.
7636 046516 112766 000055 000001  MOVVB  #'-,1(SP)     ;;MAKE THE ASCII NUMBER NEG.
7637 046524 005000      CLR    R0            ;;ZERO THE CONSTANTS INDEX
7638 046526 012703 046704      MOV    #$DBLK,R5     ;;SETUP THE OUTPUT POINTER
7639 046532 112723 000040      MOVVB  #' ,(R3)+     ;;SET THE FIRST CHARACTER TO A BLANK
7640 046536 005002      CLR    R2            ;;CLEAR THE BCD NUMBER
7641 046540 016001 046674      MOV    $DTBL(R0),R1  ;;GET THE CONSTANT
7642 046544 160105      3$:  SUB    R1,R5         ;;FORM THIS BCD DIGIT
7643 046546 002402      BLT    4$            ;;BR IF DONE
7644 046550 005202      INC    R2            ;;INCREASE THE BCD DIGIT BY 1
7645 046552 000774      BR    3$
7646 046554 060105      4$:  ADD    R1,R5         ;;ADD BACK THE CONSTANT
7647 046556 005702      TST    R2            ;;CHECK IF BCD DIGIT=0
7648 046560 001002      BNE    5$            ;;FALL THROUGH IF 0
7649 046562 105716      TSTB   (SP)          ;;STILL DOING LEADING 0'S?
7650 046564 100407      BMI    7$            ;;BR IF YES
7651 046566 106316      5$:  ASLB   (SP)          ;;MSD?
7652 046570 103003      BCC    6$            ;;BR IF NO
7653 046572 116663 00000* 177777  MOVVB  1(SP),-1(R3)  ;;YES--SET THE SIGN
7654 046600 052702 000060      6$:  BIS    #'0,R2       ;;MAKE THE BCD DIGIT ASCII
7655 046604 052702 000040      7$:  BIS    #' ,R2       ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
7656 046610 110223      MOVVB  R2,(R3)+     ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
7657 046612 005720      TST    (R0)+        ;;JUST INCREMENTING
7658 046614 020027 000010      CMP    R0,#10       ;;CHECK THE TABLE INDEX
  
```

```
7659 046620 002746          BLT      2$          ;;GO DO THE NEXT DIGIT
7660 046622 003002          BGT      8$          ;;GO TO EXIT
7661 046624 010502          MOV      R5,R2      ;;GET THE LSD
7662 046626 000764          BR       6$          ;;GO CHANGE TO ASCII
7663 046630 105726          8$: TSTB   (SP)+     ;;WAS THE LSD THE FIRST NON-ZERO?
7664 046632 100003          BPL      9$          ;;BR IF NO
7665 046634 116663 177777 177776 MOVB    -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
7666 046642 105013          9$: CLRB   (R3)      ;;SET THE TERMINATOR
7667 046644 012605          MOV     (SP)+,R5    ;;POP STACK INTO R5
7668 046646 012603          MOV     (SP)+,R3    ;;POP STACK INTO R3
7669 046650 012602          MOV     (SP)+,R2    ;;POP STACK INTO R2
7670 046652 012601          MOV     (SP)+,R1    ;;POP STACK INTO R1
7671 046654 012600          MOV     (SP)+,R0    ;;POP STACK INTO R0
7672 046656 104401 046704      TYPE    $DBLK       ;;NOW TYPE THE NUMBER
7673 046662 016666 000002 000004 MOV     2(SP),4(SP) ;;ADJUST THE STACK
7674 046670 012616          MOV     (SP)+,(SP)
7675 046672 000002          RTI                      ;;RETURN TO USER
7676 046674 023420          $DTBL: 10000.
7677 046676 001750          1000.
7678 046700 000144          100.
7679 046702 000012          10.
7680 046704 000004          $DBLK: .BLKW 4
7681                          .SBTTL TTY INPUT ROUTINE
7682
7683                          ;:*****
7684                          .ENABL LSB
7685 046714 000000          $TKCNT: .WORD 0      ;;NUMBER OF ITEMS IN QUEUE
7686 046716 000000          $TKQIN: .WORD 0     ;;INPUT POINTER
7687 046720 000000          $TKQOUT: .WORD 0    ;;OUTPUT POINTER
7688 046722 000001          $TKQSRT: .BLKB 1    ;;TTY KEYBOARD QUEUE
7689                          $TKQEND=.
7690                          .EVEN
7691
7692                          ;*TK INITIALIZE ROUTINE
7693                          ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
7694                          ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
7695                          ;:
7696                          ;*CALL:
7697                          ;*      JSR      PC,$TKINT
7698                          ;*      RETURN
7699                          ;:
7700 046724 005037 046714      $TKIN : CLR     $TKCNT      ;;CLEAR COUNT OF ITEMS IN QUEUE
7701 046730 012737 046722 046716 MOV     #$TKQSRT,$TKQIN  ;;MOVE THE STARTING ADDRESS OF THE
7702 046736 013737 046716 046720 MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
7703 046744 012737 046774 000060 MOV     #$TKSRV,@$TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
7704 046752 012737 000200 000062 MOV     #200,@$TKVEC+2  ;;"BR" LEVEL 4
7705 046760 005777 132162 TST     @$TKB           ;;CLEAR DONE FLAG
7706 046764 012777 000100 132152 MOV     #100,@$TKS     ;;ENABLE TTY KEYBOARD INTERRUPT
7707 046772 000207          RTS      PC          ;;RETURN TO CALLER
7708
7709                          ;*TK SERVICE ROUTINE
7710                          ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
7711                          ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
7712                          ;*IT IN THE QUEUE.
7713                          ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
7714                          ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)
```

```
7715  
7716 046774 117746 132146 $TKSRV: MOVB @STKB,-(SP) ;;PICKUP THE CHARACTER  
7717 047000 042716 177600 BIC #^C177,(SP) ;;STRIP THE JUNK  
7718 047004 021627 000021 CMP (SP),#$XON ;;IS IT A RANDOM XON?  
7719 047010 001002 BNE 30$ ;;BRANCH IF NO  
7720 047012 005726 TST (SP)+ ;;CLEAN RANDOM XON OFF STACK  
7721 047014 000002 RTI ;;RETURN  
7722 047016 30$:  
7723 047016 021627 000003 CMP (SP),#3 ;;IS IT A CONTROL C?  
7724 047022 001007 BNE 1$ ;;BRANCH IF NO  
7725 047024 104401 050122 TYPE ,SCNTLC ;;TYPE A CONTROL-C (^C)  
7726 047030 004737 046724 JSR PC,$TKINT ;;INIT THE KEYBOARD  
7727 047034 005726 TST (SP)+ ;;CLEAN UP STACK  
7728 047036 000137 045630 JMP CTRHLT ;;CONTROL C RESTART  
7729 047042 021627 000007 1$: CMP (SP),#7 ;;IS IT A CONTROL G?  
7730 047046 001004 BNE 2$ ;;BRANCH IF NO  
7731 047050 022737 000176 001140 CMP #SWREG,SWR ;;IS SOFT-SWR SELECTED?  
7732 047056 001500 BEQ 6$ ;;GO TO SWR CHANGE  
7733  
7734 047060 2$:  
7735 047060 022737 000001 046714 CMP #1,$TKCNT ;;IS THE QUEUE FULL?  
7736 047066 001004 BNE 3$ ;;BRANCH IF NO  
7737 047070 104401 001204 TYPE ,SBELL ;;RING THE TTY BELL  
7738 047074 005726 TST (SP)+ ;;CLEAN CHARACTER OFF OF STACK  
7739 047076 000451 BR 5$ ;;EXIT  
7740 047100 021627 000023 3$: CMP (SP),#23 ;;IS IT A CONTROL-S?  
7741 047104 001021 BNE 32$ ;;BRANCH IF NO  
7742 047106 005077 132032 CLR @STKS ;;DISABLE TTY KEYBOARD INTERRUPTS  
7743 047112 005726 TST (SP)+ ;;CLEAN CHAR OFF STACK  
7744 047114 105777 132024 51$: TSTB @STKS ;;WAIT FOR A CHAR  
7745 047120 100375 BPL 31$ ;;LOOP UNTIL ITS THERE  
7746 047122 117746 132020 MOVB @STKB,-(SP) ;;GET THE CHARACTER  
7747 047126 042716 177600 BIC #^C177,(SP) ;;MAKE IT 7-BIT ASCII  
7748 047132 022627 000021 CMP (SP)+,#21 ;;IS IT A CONTROL-Q?  
7749 047136 001366 BNE 31$ ;;BRANCH IF NO  
7750 047140 022777 000100 131776 MOV #100,@STKS ;;REENABLE TTY KEYBOARD INTERRUPTS  
7751 047146 000002 RTI ;;RETURN  
7752 047150 005237 046714 32$: INC $TKCNT ;;COUNT THIS CHARACTER  
7753 047154 021627 000140 CMP (SP),#140 ;;IS IT UPPER CASE?  
7754 047160 002405 BLT 4$ ;;BRANCH IF YES  
7755 047162 021627 000175 CMP (SP),#175 ;;IS IT A SPECIAL CHAR?  
7756 047166 003002 BGT 4$ ;;BRANCH IF YES  
7757 047170 042716 000040 BIC #40,(SP) ;;MAKE IT UPPER CASE  
7758 047174 112677 177516 MOVB (SP)+,@STKQIN ;;AND PUT IT IN QUEUE  
7759 047200 005237 046716 INC $TKQIN ;;UPDATE THE POINTER  
7760 047204 023727 046716 046723 CMP $TKQIN,$$TKQEND ;;GO OFF THE END?  
7761 047212 001003 BNE 5$ ;;BRANCH IF NO  
7762 047214 012737 046722 046716 MOV #$$TKQRT,$TKQIN ;;RESET THE POINTER  
7763 047222 000002 5$: RTI ;;RETURN  
7764  
7765  
7766  
7767  
7768  
7769  
7770 047224 022737 000176 001140 $CKSWR: CMP #SWREG,SWR ;;IS THE SOFT-SWR SELECTED
```

```
7771 047232 001124          BNE      15$          ;;EXIT IF NOT
7772 047234 105777 131704   TSTB     @STKS       ;;IS A CHAR WAITING?
7773 047240 100121          BPL      15$          ;;IF NOT, EXIT
7774 047242 117746 131700   MOVB     @STKB,-(SP)  ;;YES
7775 047246 042716 177600   BIC      #^C177,(SP) ;;MAKE IT 7-BIT ASCII
7776 047252 021627 000C07   CMP      (SP),#7     ;;IS IT A CONTROL-G?
7777 047256 001300          BNE      2$          ;;IF NOT, PUT IT IN THE TTY QUEUE
7778                                     ;;AND EXIT
7779
7780                                     ;:*****
7781                                     ;:*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
7782                                     ;:*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
7783                                     ;:*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
7784 047260 123727 001134 000001 6$:  CMPB     $AUTOB,#1  ;;ARE WE RUNNING IN AUTO-MODE?
7785 047266 001674          BEQ      2$          ;;BRANCH IF YES
7786 047270 005726          TST      (SP)+       ;;CLEAR CONTROL-G OFF STACK
7787 047272 004737 046724   JSR      PC,$TKINT   ;;FLUSH THE TTY INPUT QUEUE
7788 047276 005077 131642   CLR      @STKS       ;;DISABLE TTY KEYBOARD INTERRUPTS
7789 047302 112737 000001 001135   MOVB     #1,$INTAG   ;;SET INTERRUPT MODE INDICATOR
7790
7791 047310 104401 050134          TYPE     ,$CNTLG     ;;ECHO THE CONTROL-G (^G)
7792 047314 104401 050141   $GTSWR: TYPE     ,$MSWR  ;;TYPE CURRENT CONTENTS
7793 047320 013746 000176   MOV      SWREG,-(SP) ;;SAVE SWREG FOR TYPEOUT
7794 047324 104402          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7795 047326 104401 050152          TYPE     ,$MNEW     ;;PROMPT FOR NEW SWR
7796 047332 005046          CLR      -(SP)      ;;CLEAR COUNTER
7797 047334 005046          CLR      -(SP)      ;;THE NEW SWR
7798 047336 105777 131602   7$:  TSTB     @STKS       ;;CHAR THERE?
7799 047342 100375          BPL      7$          ;;IF NOT TRY AGAIN
7800
7801 047344 117746 131576   MOVB     @STKB,-(SP) ;;PICK UP CHAR
7802 047350 042716 177600   BIC      #^C177,(SP) ;;MAKE IT 7-BIT ASCII
7803
7804 047354 021627 000003          CMP      (SP),#3    ;;IS IT A CONTROL-C?
7805 047360 001015          BNE      9$          ;;BRANCH IF NOT
7806 047362 104401 050122          TYPE     ,$CNTLC    ;;YES, ECHO CONTROL-C (^C)
7807 047366 062706 000006          ADD      #6,SP      ;;CLEAN UP STACK
7808 047372 123727 001135 000001   CMPB     $INTAG,#1  ;;REENABLE TTY KEYBOARD INTERRUPTS?
7809 047400 001003          BNE      8$          ;;BRANCH IF NO
7810 047402 012777 000100 131534   MOV      #100,@STKS ;;ALLOW TTY KEYBOARD INTERRUPTS
7811 047410 000137 045630   8$:  JMP      CTRHLT     ;;CONTROL-C RESTART
7812
7813
7814 047414 021627 000025   9$:  CMP      (SP),#25   ;;IS IT A CONTROL-U?
7815 047420 001005          BNE      10$         ;;BRANCH IF NOT
7816 047422 104401 050127          TYPE     ,$CNTLU    ;;YES, ECHO CONTROL-U (^U)
7817 047426 062706 000006   20$: ADD      #6,SP      ;;IGNORE PREVIOUS INPUT
7818 047432 000737          BR      19$         ;;LET'S TRY IT AGAIN
7819
7820
7821 047434 021627 000015   10$: CMP      (SP),#15   ;;IS IT A <CR>?
7822 047440 001022          BNE      16$         ;;BRANCH IF NO
7823 047442 005766 000004          TST      4(SP)      ;;YES, IS IT THE FIRST CHAR?
7824 047446 001403          BEQ      11$         ;;BRANCH IF YES
7825 047450 016677 000002 131462   MOV      2(SP),@SWR  ;;SAVE NEW SWR
7826 047456 062706 000006   11$: ADD      #6,SP      ;;CLEAR UP STACK
```

```
7827 047462 104401 001211 14$: TYPE $CRLF ;;ECHO <CR> AND <LF>
7828 047466 123727 001135 000001 CMPB $INTAC,#1 ;;RE-ENABLE TTY KBD INTERRUPTS?
7829 047474 001003 BNE 15$ ;;BRANCH IF NOT
7830 047476 012777 000100 131440 MOV #100,@$TKS ;;RE-ENABLE TTY KBD INTERRUPTS
7831 047504 000002 15$: RTI ;;RETURN
7832 047506 004737 046120 16$: JSR PC,$TYPEC ;;ECHO CHAR
7833 047512 021627 000060 CMP (SP),#60 ;;CHAR < 0?
7834 047516 002420 BLT 18$ ;;BRANCH IF YES
7835 047520 021627 000067 CMP (SP),#67 ;;CHAR > ??
7836 047524 003015 BGT 18$ ;;BRANCH IF YES
7837 047526 042726 000060 BIC #60,(SP)+ ;;STRIP-OFF ASCII
7838 047532 005766 000002 TST 2(SP) ;;IS THIS THE FIRST CHAR
7839 047536 001403 BEQ 17$ ;;BRANCH IF YES
7840 047540 006316 ASL (SP) ;;NO, SHIFT PRESENT
7841 047542 006316 ASL (SP) ;; CHAR OVER TO MAKE
7842 047544 006316 ASL (SP) ;; ROOM FOR NEW ONE.
7843 047546 005266 000002 17$: INC 2(SP) ;;KEEP COUNT OF CHAR
7844 047552 056616 177776 BIS -2(SP),(SP) ;;SET IN NEW CHAR
7845 047556 000667 BR 7$ ;;GET THE NEXT ONE
7846 047560 104401 001210 18$: TYPE $QUES ;;TYPE ?<CR><LF>
7847 047564 000720 BR 20$ ;;SIMULATE CONTROL-U
7848 .DSABL LSB
7849
7850
7851
7852 *****
7853 *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
7854 *CALL:
7855 * RDCHR ;;GET A CHARACTER FROM THE QUEJE
7856 * RETURN HERE ;;CHARACTER IS ON THE STACK
7857 * ;;WITH PARITY BIT STRIPPED OFF
7858
7859 $RDCHR: MOV (SP),-(SP) ;;PUSH DOWN THE PC AND
7860 047570 016666 000004 000002 MOV 4(SP),2(SP) ;;THE PS
7861 047576 005066 000004 CLR 4(SP) ;;GET READY FOR A CHARACTER
7862 047602 005046 CLR -(SP) ;;PUT NEW PS ON STACK
7863 047604 012746 047612 MOV #64$,-(SP) ;;PUT NEW PC ON STACK
7864 047610 000002 RTI ;;POP NEW PC AND PS
7865 047612 64$:
7866 047612 005737 046714 1$: TST $TKCNT ;;WAIT ON A CHARACTER
7867 047616 001775 BEQ 1$
7868 047620 005337 046714 DEC $TKCNT ;;DECREMENT THE COUNTER
7869 047624 117766 177070 000004 MOVB @$TKQOUT,4(SP) ;;GET ONE CHARACTER
7870 047632 005237 046720 INC $TKQOUT ;;UPDATE THE POINTER
7871 047636 023727 046720 046723 CMP $TKQOUT,$$TKQEND ;;DID IT GO OFF OF THE END?
7872 047644 001003 BNE 2$ ;;BRANCH IF NO
7873 047646 012737 046722 046720 MOV #$$TKQSRIT,$$TKQOUT ;;RESET THE POINTER
7874 047654 000002 2$: RTI ;;RETURN
7875 *****
7876 *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
7877 *CALL:
7878 * RDLIN ;;INPUT A STRING FROM THE TTY
7879 * RETURN HERE ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
7880 * ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
7881
7882 047656 010346 $RDLIN: MOV R3,-(SP) ;;SAVE R3
```

7883	047660	005046			CLR	-(SP)	::CLEAR THE RUBOUT KEY
7884	047662	012703	050112		1\$: MOV	#\$TTYIN,R3	::GET ADDRESS
7885	047666	022703	050122		2\$: CMP	#\$TTYIN+8.,R3	::BUFFER FULL?
7886	047672	101456			BLOS	4\$	::BR IF YES
7887	047674	104410			RDCHR		::GO READ ONE CHARACTER FROM THE TTY
7888	047676	112613			MOVB	(SP)+,(R3)	::GET CHARACTER
7889	047700	122713	000177		10\$: CMPB	#177,(R3)	::IS IT A RUBOUT
7890	047704	001022			BNE	5\$	::BR IF NO
7891	047706	005716			TST	(SP)	::IS THIS THE FIRST RUBOUT?
7892	047710	001007			BNE	6\$	::BR IF NO
7893	047712	112737	000134	050110	MOVB	#'\,9\$	::TYPE A BACK SLASH
7894	047720	104401	050110		TYPE	,9\$	
7895	047724	012716	177777		MOV	#-1,(SP)	::SET THE RUBOUT KEY
7896	047730	005303			6\$: DEC	R3	::BACKUP BY ONE
7897	047732	020327	050112		CMP	R3,\$TTYIN	::STACK EMPTY?
7898	047736	103434			BLO	4\$	::BR IF YES
7899	047740	111337	050110		MOVB	(R3),9\$	::SETUP TO TYPEOUT THE DELETED CHAR.
7900	047744	104401	050110		TYPE	,9\$	::GO TYPE
7901	047750	000746			BR	2\$	::GO READ ANOTHER CHAR.
7902	047752	005716			5\$: TST	(SP)	::RUBOUT KEY SET?
7903	047754	001406			BFO	7\$	::BR IF NO
7904	047756	112737	000134	050110	MOVB	#'\,9\$	::TYPE A BACK SLASH
7905	047764	104401	050110		TYPE	,9\$	
7906	047770	005016			CLR	(SP)	::CLEAR THE RUBOUT KEY
7907	047772	122713	000025		7\$: CMPB	#25,(R3)	::IS CHARACTER A CTRL U?
7908	047776	001003			BNE	8\$	::BR IF NO
7909	050000	104401	050127		TYPE	,\$CNTLU	::TYPE A CONTROL 'U'
7910	050004	000726			BR	1\$	::GO START OVER
7911	050006	122713	000022		8\$: CMPB	#22,(R3)	::IS CHARACTER A '^R'?
7912	050012	001011			BNE	3\$	::BRANCH IF NO
7913	050014	105013			CLRB	(R3)	::CLEAR THE CHARACTER
7914	050016	104401	001211		TYPE	,\$CRLF	::TYPE A 'CR' & 'LF'
7915	050022	104401	050112		TYPE	,\$TTYIN	::TYPE THE INPUT STRING
7916	050026	000717			BR	2\$	::GO PICKUP ANOTHER CHARACTER
7917	050030	104401	001210		4\$: TYPE	,\$QUES	::TYPE A '?'
7918	050034	000712			BR	1\$	::CLEAR THE BUFFER AND LOOP
7919	050036	111337	050110		3\$: MOVB	(R3),9\$	::ECHO THE CHARACTER
7920	050042	104401	050110		TYPE	,9\$	
7921	050046	122723	000015		CMPB	#15,(R3)+	::CHECK FOR RETURN
7922	050052	001305			BNE	2\$	::LOOP IF NOT RETURN
7923	050054	105063	177777		CLRB	-1(R3)	::CLEAR RETURN (THE 15)
7924	050060	104401	001212		TYPE	,\$LF	::TYPE A LINE FEED
7925	050064	005726			TST	(SP)+	::CLEAN RUBOUT KEY FROM THE STACK
7926	050066	012603			MOV	(SP)+,R3	::RESTORE R3
7927	050070	011646			MOV	(SP)-,(SP)	::ADJUST THE STACK AND PUT ADDRESS OF THE
7928	050072	016666	000004	000002	MOV	4(SP),2(SP)	::FIRST ASCII CHARACTER ON IT
7929	050100	012766	050112	000004	MOV	#\$TTYIN,4(SP)	
7930	050106	000002			RTI		::RETURN
7931	050110	000			9\$: .BYTE	0	::STORAGE FOR ASCII CHAR. TO TYPE
7932	050111	000			.BYTE	0	::TERMINATOR
7933	050112	000010			\$.BLKB	8.	::RESERVE 8 BYTES FOR TTY INPUT
7934	050122	041536	005015	000	\$.ASCIZ	/'^C/<15><12>	::CONTROL '^C'
7935	050127	136	006525	000012	\$.ASCIZ	/'^U/<15><12>	::CONTROL '^U'
7936	050134	043536	005015	000	\$.ASCIZ	/'^G/<15><12>	::CONTROL '^G'
7937	050141	015	051412	051127	\$.MSWR:	<15><12>/SWR = /	
7938	050146	036440	000040				

```

7939 050152 020040 042516 020127 $MNEW: .ASCIZ / NEW = /
7940 050160 020075 000
7941 050164
7942 .EVEN
7943 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
7944
7945 ::*****
7946 :*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
7947 :*CHANGE IT TO BINARY.
7948 :*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
7949 :*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
7950 :*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
7951 :*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
7952 :*CALL:
7953 :*      RDOCT          ;;READ AN OCTAL NUMBER
7954 :*      RETURN HERE   ;;LOW ORDER BITS ARE ON TOP OF THE STACK
7955 :*                   ;;HIGH ORDER BITS ARE IN $HI OCT
7956 050164 011646          $RDOCT: MOV      (SP),-(SP)   ;;PROVIDE SPACE FOR THE
7957 050166 016666 000004 000002 MOV      4(SP),2(SP)   ;;INPUT NUMBER
7958 050174 010046          MOV      R0,-(SP)    ;;PUSH R0 ON STACK
7959 050176 010146          MOV      R1,-(SP)    ;;PUSH R1 ON STACK
7960 050200 010246          MOV      R2,-(SP)    ;;PUSH R2 ON STACK
7961 050202 104411          1$:  RDLIN          ;;READ AN ASCIZ LINE
7962 050204 012600          MOV      (SP)+,R0    ;;GET ADDRESS OF 1ST CHARACTER
7963 050206 010037 050312 MOV      R0,5$        ;;AND SAVE IT
7964 050212 005001          CLR      R1          ;;CLEAR DATA WORD
7965 050214 005002          CLR      R2
7966 050216 112046          2$:  MOVB     (R0)+,-(SP)   ;;PICKUP THIS CHARACTER
7967 050220 001420          BEQ     3$          ;;IF ZERO GET OUT
7968 050222 122716 000060 CMPB    #'0,(SP)      ;;MAKE SURE THIS CHARACTER
7969 050226 003026          BGT     4$          ;;IS AN OCTAL DIGIT
7970 050230 122716 000067 CMPB    #'7,(SP)
7971 050234 002423          BLT     4$
7972 050236 006301          ASL     R1          ;;*2
7973 050240 006102          ROL     R2
7974 050242 006301          ASL     R1          ;;*4
7975 050244 006102          ROL     R2
7976 050246 006301          ASL     R1          ;;*8
7977 050250 006102          ROL     R2
7978 050252 042716 177770 BIC     #'^C7,(SP)   ;;STRIP THE ASCII JUNK
7979 050256 062601          ADD     (SP)+,R1    ;;ADD IN THIS DIGIT
7980 050260 000756          BR      2$          ;;LOOP
7981 050262 005726          3$:  TST     (SP)+    ;;CLEAN TERMINATOR FROM STACK
7982 050264 010166 000012 MOV     R1,12(SP)    ;;SAVE THE RESULT
7983 050270 010237 050322 MOV     R2,$HI OCT
7984 050274 012602          MOV     (SP)+,R2    ;;POP STACK INTO R2
7985 050276 012601          MOV     (SP)+,R1    ;;POP STACK INTO R1
7986 050300 012600          MOV     (SP)+,R0    ;;POP STACK INTO R0
7987 050302 000002          RTI          ;;RETURN
7988 050304 005726          4$:  TST     (SP)+    ;;CLEAN PARTIAL FROM STACK
7989 050306 105010          CLRB   (R0)        ;;SET A TERMINATOR
7990 050310 104401          TYPE          ;;TYPE UP THRU THE BAD CHAR.
7991 050312 000000          5$:  .WORD   0
7992 050314 104401 001210 TYPE    $QUES
7993 050320 000730          BR     1$          ;;TRY AGAIN
7994 050322 000000          $HI OCT: .WORD 0   ;;HIGH ORDER BITS GO HERE

```

```

7995 .SBTTL SAVE AND RESTORE R0-R5 ROUTINES
7996
7997 :*****
7998 :*SAVE R0-R5
7999 :*CALL:
8000 :* SAVREG
8001 :*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
8002 :*
8003 :*TOP---(+16)
8004 :* +2---(+18)
8005 :* +4---R5
8006 :* +6---R4
8007 :* +8---R3
8008 :*+10---R2
8009 :*+12---R1
8010 :*+14---R0
8011
8012 050324 $SAVREG:
8013 050324 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
8014 050326 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
8015 050330 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
8016 050332 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
8017 050334 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
8018 050336 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
8019 050340 016646 000022 MOV 22(SP),-(SP) ;;SAVE PS OF MAIN FLOW
8020 050344 016646 000022 MOV 22(SP),-(SP) ;;SAVE PC OF MAIN FLOW
8021 050350 016646 000022 MOV 22(SP),-(SP) ;;SAVE PS OF CALL
8022 050354 016646 000022 MOV 22(SP),-(SP) ;;SAVE PC OF CALL
8023 050360 000002 RTI
8024
8025 :*RESTORE R0-R5
8026 :*CALL:
8027 :* RESREG
8028 050362 $RESREG:
8029 050362 012666 000022 MOV (SP)+,22(SP) ;;RESTORE PC OF CALL
8030 050366 012666 000022 MOV (SP)+,22(SP) ;;RESTORE PS OF CALL
8031 050372 012666 000022 MOV (SP)+,22(SP) ;;RESTORE PC OF MAIN FLOW
8032 050376 012666 000022 MOV (SP)+,22(SP) ;;RESTORE PS OF MAIN FLOW
8033 050402 012605 MOV (SP)+,R5 ;;POP STACK INTO R5
8034 050404 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
8035 050406 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
8036 050410 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
8037 050412 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
8038 050414 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
8039 050416 000002 RTI
8040
8041 .SBTTL POWER DOWN AND UP ROUTINE
8042
8043 :*****
8044
8045 :POWER DOWN ROUTINE
8046 050420 017737 130514 003634 $PWRDN: MOV @SWR,SAVSWR ;SAVE SWITCH REGISTER
8047 050426 012737 050446 000024 MOV #PWRUP,PWRVEC ;SET JP VECTOR
8048 050434 012737 000340 000026 MOV #PR7,PWRVEC+2
8049 050442 000000 HALT
8050 050444 000776 BR .-2 ;HANG UP
  
```



```
8051
8052
8053
8054 :POWER UP ROUTINE
8055 050446 005037 050536 $PWRUP: CLR $PWRCT ;LOAD WAIT COUNT
8056 050452 012737 000144 050540 MOV #100,$PWRCT+2
8057 050460 005237 050536 1$: INC $PWRCT ;WAIT FOR TELETYPE
8058 050464 001375 BNE 1$
8059 050466 005337 050540 DEC $PWRCT+2
8060 050472 001372 BNE 1$
8061 050474 012737 050420 000024 MOV #SPWRDN,PWRVEC ;SET UP FOR POWER DOWN VECTOR
8062 050502 012737 000340 000026 MOV #PR7,PWRVEC+2
8063 050510 012706 001100 MOV #STACK,SP ;FORCE STACK
8064 050514 104401 050542 TYPE $POWER ;TYPE POWER
8065 050520 013777 003634 130412 MOV SAVSWR,@SWR ;RESTORE SWITCH REGISTER
8066 050526 013702 001270 MOV $BASE,R2 ;REINITIALISE R2 FOR '611 BASE
8067 050532 000177 130350 JMP @SLPADR ;GO BACK TO LAST TEST
8068
8069 050536 000000 000000 $PWRCT: .WORD 0,0 ;TELETYPE TIME OUT
8070 050542 047520 042527 000122 $POWER: .ASCIZ /POWER/
8071 .FVEN
8072 .SBTTL TRAP DECODER
8073
8074
8075 :*****
8076 :*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
8077 :*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
8078 :*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
8079 :*GO TO THAT ROUTINE.
8080 050550 010046 $TRAP: MOV R0,-(SP) ;;SAVE R0
8081 050552 016600 000002 MOV 2(SP),R0 ;;GET TRAP ADDRESS
8082 050556 005740 TST -(R0) ;;BACKUP BY 2
8083 050560 111000 MOV#B (R0),R0 ;;GET RIGHT BYTE OF TRAP
8084 050562 006300 ASL R0 ;;POSITION FOR INDEXING
8085 050564 016000 050604 MOV $TRPAD(R0),R0 ;;INDEX TO TABLE
8086 050570 000200 RTS R0 ;;GO TO ROUTINE
8087
8088
8089 ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
8090
8091 050572 011646 $TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
8092 050574 016666 000004 000002 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
8093 050602 000002 RTI ;;RESTORE THE PSW
8094
8095 .SBTTL TRAP TABLE
8096
8097 :*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
8098 :*BY THE "TRAP" INSTRUCTION.
8099
8100 : ROUTINE
8101 :-----
8102 050604 050572 $TRPAD: .WORD $TRAP2
8103 050606 045706 $TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
8104 050610 046266 $TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
8105 050612 046242 $TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
8106 050614 046302 $TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
```

8107	050616	046470	\$TYPDS	::CALL=TYPDS	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)
8108						
8109	050620	047314	\$GTSWR	::CALL=GTSWR	TRAP+6(104406)	GET SOFT-SWR SETTING
8110						
8111	050622	047224	\$CKSWR	::CALL=CKSWR	TRAP+7(104407)	TEST FOR CHANGE IN SOFT-SWR
8112	050624	047566	\$RDCHR	::CALL=RDCHR	TRAP+10(104410)	TTY TYPEIN CHARACTER ROUTINE
8113	050626	047656	\$RDLIN	::CALL=RDLIN	TRAP+11(104411)	TTY TYPEIN STRING ROUTINE
8114	050630	050164	\$RDOCT	::CALL=RDOCT	TRAP+12(104412)	READ AN OCTAL NUMBER FROM TTY
8115	050632	050324	\$SAVREG	::CALL=SAVREG	TRAP+13(104413)	SAVE R0-R5 ROUTINE
8116	050634	050362	\$RESREG	::CALL=RESREG	TRAP+14(104414)	RESTORE R0-R5 ROUTINE
8117	050636	044654	\$SCOP1\$	::CALL=SCOP1	TRAP+15(104415)	INTERNAL LOOP ON ERROR

Line	Code	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6	Field 7	Field 8
8118						.SBTTL	DATA TABLE FOR PRINT OUT		
8119									
8120	050640	001220	003604			DT000:	.WORD	\$TESTN,TRAPPC	
8121	050644	001220	001116	003500		DT001:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.MR2,T.MR2,E.MR3,T.MR3	
8122	050652	003440	003526	003466					
8123	050660	003530	003470						
8124	050664	001220	001116	003500		DT015:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1	
8125	050672	003440							
8126	050674	001220	001116	003524		DT017:	.WORD	\$TESTN,\$ERRPC,E.MR1,T.MR1,PR.BIT,M1.BIT,BITCNT	
8127	050702	003464	003614	003616					
8128	050710	003622							
8129	050712	001220	001116	003500		DT022:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.BA,T.BA,E.WC,T.WC	
8130	050720	003440	003504	003444					
8131	050726	003502	003442						
8132	050732	001220	001116	003524		DT025:	.WORD	\$TESTN,\$ERRPC,E.MR1,T.MR1,BITCNT	
8133	050740	003464	003622						
8134	050744	001220	001116	003500		DT031:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.BA,T.BA,E.WC,T.WC,E.MR1,T.MR1	
8135	050752	003440	003504	003444					
8136	050760	003502	003442	003524					
8137	050766	003464							
8138	050770	001220	001116	003500		DT035:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.BA,T.BA	
8139	050776	003440	003510	003450					
8140	051004	003504	003444						
8141	051010	003502	003442						
8142	051014	001220	001116	003522		DT041:	.WORD	E.WC,T.WC \$TESTN,\$ERRPC,E.DB,T.DB	
8143	051022	003462							
8144	051024	001220	001116	003500		DT042:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2	
8145	051032	003440	003510	003450					
8146	051040	001220	001116	003522		DT054:	.WORD	\$TESTN,\$ERRPC,E.DB,T.DB,WRDCNT	
8147	051046	003462	003624						
8148	051052	001220	001116	003500		DT055:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,WRDCNT	
8149	051060	003440	003510	003450					
8150	051066	003624							
8151	051070	001220	001116	003500		DT072:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.ER,T.FR	
8152	051076	003440	003510	003450					
8153	051104	003514	003454						
8154	051110	003504	003444	003502			.WORD	E.BA,T.BA,E.WC,T.WC	
8155	051116	003442							
8156	051120	001220	001116	003500		DT077:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2	
8157	051126	003440	003510	003450					
8158	051134	001220	001116	003500		DT150:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,WRDCNT,BITCNT	
8159	051142	003440	003624	003622					
8160	051150	001220	001116	003500		DT164:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS1,BITCNT	
8161	051156	003440	003510	003440					
8162	051164	003622							
8163	051166	001220	001116	003524		DT170:	.WORD	\$TESTN,\$ERRPC,E.MR1,T.MR1,P1.BIT,PR.BIT,M1.BIT,M2.BIT,BITCNT,SECCNT	
8164	051174	003464	003612	003614					
8165	051202	003616	003620	003622					
8166	051210	003626							
8167	051212	001220	001116	003524		DT171:	.WORD	\$TESTN,\$ERRPC,E.MR1,T.MR1	
8168	051220	003464							
8169	051222	001220	001116	003500		DT175:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.DS,T.DS,E.ER,T.ER	
8170	051230	003440	003510	003450					
8171	051236	003512	003452	003514					
8172	051244	003454							
8173	051246	001220	001116	003500		DT211:	.WORD	\$TESTN,\$ERRPC,E.CS1,T.CS1,E.CS2,T.CS2,E.DS,T.DS,E.ER,T.ER	

8174	051254	003440	003510	003450
8175	051262	003512	003452	003514
8176	051270	003454		
8177	051272	003540	003550	003552
8178	051300	003554		

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

8179  
8180  
8181 051302 000001  
8182 051304 002 000  
8183 051306 000005  
8184 051310 000 000  
8185 051312 052467  
8186 051314 000 000  
8187 051316 052505  
8188 051320 002 000  
8189 051322 052551  
8190 051324 000 000  
8191 051326 052630  
8192 051330 006 000  
8193 051332 000005  
8194 051334 000 000  
8195 051336 052467  
8196 051340 000 000  
8197 051342 052505  
8198 051344 002 000  
8199 051346 052707  
8200 051350 000 000  
8201 051352 052726  
8202 051354 002 000  
8203 051356 000005  
8204 051360 000 000  
8205 051362 052467  
8206 051364 000 000  
8207 051366 052505  
8208 051370 002 000  
8209 051372 052744  
8210 051374 000 000  
8211 051376 053010  
8212 051400 005 000  
8213 051402 000005  
8214 051404 000 000  
8215 051406 052467  
8216 051410 000 000  
8217 051412 052505  
8218 051414 002 000  
8219 051416 053056  
8220 051420 000 000  
8221 051422 053135  
8222 051424 006 000  
8223 051426 000005  
8224 051430 000 000  
8225 051432 052467  
8226 051434 000 000  
8227 051436 052505  
8228 051440 002 000  
8229 051442 053212  
8230 051444 000 000  
8231 051446 053236  
8232 051450 003 000  
8233 051452 000005  
8234 051454 000 000

.SBTTL DATA FORMAT FOR PRINT OUT  
DF000: .WORD 1  
.BYTE 2,0  
DF001: .WORD 5 ;ERRORS 1-14  
.BYTE 0,0  
.WORD DH000A  
.BYTE 0,0  
.WORD DH000B  
.BYTE 2,0  
.WORD DH001A  
.BYTE 0,0  
.WORD DH001B  
.BYTE 6,0  
DF015: .WORD 5 ;ERRORS 15-16  
.BYTE 0,0  
.WORD DH000A  
.BYTE 0,0  
.WORD DH000B  
.BYTE 2,0  
.WORD DH015A  
.BYTE 0,0  
.WORD DH015B  
.BYTE 2,0  
DF017: .WORD 5 ;ERROR 17-21  
.BYTE 0,0  
.WORD DH000A  
.BYTE 0,0  
.WORD DH000B  
.BYTE 2,0  
.WORD DH017A  
.BYTE 0,0  
.WORD DH017B  
.BYTE 5,0  
DF022: .WORD 5 ;ERROR 22-24  
.BYTE 0,0  
.WORD DH000A  
.BYTE 0,0  
.WORD DH000B  
.BYTE 2,0  
.WORD DH022A  
.BYTE 0,0  
.WORD DH022B  
.BYTE 6,0  
DF025: .WORD 5 ;ERROR 25  
.BYTE 0,0  
.WORD DH000A  
.BYTE 0,0  
.WORD DH000B  
.BYTE 2,0  
.WORD DH025A  
.BYTE 0,0  
.WORD DH025B  
.BYTE 3,0  
DF031: .WORD 5 ;ERROR 31-34  
.BYTE 0,0

8235	051456	052467		.WORD	DH000A	
8236	051460	000	000	.BYTE	0,0	
8237	051462	052505		.WORD	DH000B	
8238	051464	002	000	.BYTE	2,0	
8239	051466	053264		.WORD	DH031A	
8240	051470	000	000	.BYTE	0,0	
8241	051472	053363		.WORD	DH031B	
8242	051474	010	000	.BYTE	8.,0	
8243	051476	000005		.WORD	5	DF035: ;ERRORS 35-40
8244	051500	000	000	.BYTE	0,0	
8245	051502	052467		.WORD	DH000A	
8246	051504	000	000	.BYTE	0,0	
8247	051506	052505		.WORD	DH000B	
8248	051510	002	000	.BYTE	2,0	
8249	051512	053461		.WORD	DH035A	
8250	051514	000	000	.BYTE	0,0	
8251	051516	053560		.WORD	DH035B	
8252	051520	010	000	.BYTE	8.,0	
8253	051522	000005		.WORD	5	DF041: ;ERROR 41
8254	051524	000	000	.BYTE	0,0	
8255	051526	052467		.WORD	DH000A	
8256	051530	000	000	.BYTE	0,0	
8257	051532	052505		.WORD	DH000B	
8258	051534	002	000	.BYTE	2,0	

8259	051536	053655		.WORD	DH041A	
8260	051540	000	000	.BYTE	0.0	
8261	051542	053672		.WORD	DH041B	
8262	051544	002	000	.BYTE	2.0	
8263	051546	000005		DF042: .WORD	5	;ERRORS 42-43
8264	051550	000	000	.BYTE	0.0	
8265	051552	052467		.WORD	DH000A	
8266	051554	000	000	.BYTE	0.0	
8267	051556	052505		.WORD	DH000B	
8268	051560	002	000	.BYTE	2.0	
8269	051562	053707		.WORD	DH042A	
8270	051564	000	000	.BYTE	0.0	
8271	051566	053746		.WORD	DH042B	
8272	051570	004	000	.BYTE	4.0	
8273	051572	000005		DF054: .WORD	5	;ERROR 54
8274	051574	000	000	.BYTE	0.0	
8275	051576	052467		.WORD	DH000A	
8276	051600	000	000	.BYTE	0.0	
8277	051602	052505		.WORD	DH000B	
8278	051604	002	000	.BYTE	2.0	
8279	051606	054004		.WORD	DH054A	
8280	051610	000	000	.BYTE	0.0	
8281	051612	054031		.WORD	DH054B	
8282	051614	003	000	.BYTE	3.0	
8283	051616	000005		DF055: .WORD	5	;ERROR 55
8284	051620	000	000	.BYTE	0.0	
8285	051622	052467		.WORD	DH000A	
8286	051624	000	000	.BYTE	0.0	
8287	051626	052505		.WORD	DH000B	
8288	051630	002	000	.BYTE	2.0	
8289	051632	054057		.WORD	DH055A	
8290	051634	000	000	.BYTE	0.0	
8291	051636	054124		.WORD	DH055B	
8292	051640	005	000	.BYTE	5.0	
8293	051642	000007		DF072: .WORD	7	
8294	051644	000	000	.BYTE	0.0	
8295	051646	052467		.WORD	DH000A	
8296	051650	000	000	.BYTE	0.0	
8297	051652	052505		.WORD	DH000B	
8298	051654	002	000	.BYTE	2.0	
8299	051656	054172		.WORD	DH072A	
8300	051660	000	000	.BYTE	0.0	
8301	051662	054231		.WORD	DH072B	
8302	051664	004	000	.BYTE	4.0	
8303	051666	054267		.WORD	DH072C	
8304	051670	000	000	.BYTE	0.0	
8305	051672	054346		.WORD	DH072D	
8306	051674	006	000	.BYTE	6.0	
8307	051676	000007		DF077: .WORD	7	
8308	051700	000	000	.BYTE	0.0	
8309	051702	052467		.WORD	DH000A	
8310	051704	000	000	.BYTE	0.0	
8311	051706	052505		.WORD	DH000B	
8312	051710	002	000	.BYTE	2.0	
8313	051712	054172		.WORD	DH072A	
8314	051714	000	000	.BYTE	0.0	

8315	051716	054231			.WORD	DH072B		
8316	051720	004	000		.BYTE	4.0		
8317	051722	000005		DF150:	.WORD	5		;ERROR 150
8318	051724	000	000		.BYTE	0.0		
8319	051726	052467			.WORD	DH000A		
8320	051730	000	000		.BYTE	0.0		
8321	051732	052505			.WORD	DH000B		
8322	051734	002	000		.BYTE	2.0		
8323	051736	054423			.WORD	DH150A		
8324	051740	000	000		.BYTE	0.0		
8325	051742	054457			.WORD	DH150B		
8326	051744	004	000		.BYTE	4.0		
8327	051746	000005		DF164:	.WORD	5		;ERROR 164
8328	051750	000	000		.BYTE	0.0		
8329	051752	052467			.WORD	DH000A		
8330	051754	000	000		.BYTE	0.0		
8331	051756	052505			.WORD	DH000B		
8332	051760	002	000		.BYTE	2.0		
8333	051762	054515			.WORD	DH164A		
8334	051764	000	000		.BYTE	0.0		
8335	051766	054561			.WORD	DH164B		
8336	051770	005	000		.BYTE	5.0		
8337	051772	000005		DF170:	.WORD	5		;ERROR 170
8338	051774	000	000		.BYTE	0.0		
8339	051776	052467			.WORD	DH000A		
8340	052000	000	000		.BYTE	0.0		
8341	052002	052505			.WORD	DH000B		
8342	052004	002	000		.BYTE	2.0		
8343	052006	054627			.WORD	DH170A		
8344	052010	000	000		.BYTE	0.0		
8345	052012	054726			.WORD	DH170B		
8346	052014	010	000		.BYTF	8..0		
8347	052016	000005		DF171:	.WORD	5		;ERRORS 171-174
8348	052020	000	000		.BYTE	0.0		
8349	052022	052467			.WORD	DH000A		
8350	052024	000	000		.BYTE	0.0		
8351	052026	052505			.WORD	DH000B		
8352	052030	002	000		.BYTE	2.0		
8353	052032	055024			.WORD	DH171A		
8354	052034	000	000		.BYTE	0.0		
8355	052036	055043			.WORD	DH171B		
8356	052040	002	000		.BYTE	2.0		
8357	052042	000005		DF175:	.WORD	5		;ERRORS 175-210
8358	052044	000	000		.BYTE	0.0		
8359	052046	052467			.WORD	DH000A		
8360	052050	000	000		.BYTE	0.0		
8361	052052	052505			.WORD	DH000B		
8362	052054	002	000		.BYTE	2.0		
8363	052056	055061			.WORD	DH175A		
8364	052060	000	000		.BYTE	0.0		
8365	052062	055160			.WORD	DH175B		
8366	052064	010	000		.BYTE	8..0		
8367	052066	000007		DF211:	.WORD	7		;ERRORS 211-214
8368	052070	000	000		.BYTE	0.0		
8369	052072	052467			.WORD	DH000A		
8370	052074	000	000		.BYTE	0.0		



CZR6CEO RK611 DSKLS CTRL PRT3  
CZR6CE.P11 27-AUG-81 10:24

MACY11 30(1046) 28-AUG-81 10:35  
DATA FORMAT FOR PRINT OUT

B 13  
PAGE 158

SEQ 0157

8371	052076	052505		.WORD	DH000B
8372	052100	002	000	.BYTE	2,0
8373	052102	055061		.WORD	DH175A
8374	052104	000	000	.BYTE	0,0
8375	052106	055160		.WORD	DH175B
8376	052110	010	000	.BYTE	8,0
8377	052112	055255		.WORD	DH211A
8378	052114	000	000	.BYTE	0,0
8379	052116	055303		.WORD	DH211B
8380	052120	004	000	.BYTE	4,0

```
8381 .SBTTL ASCII MESSAGES
8382
8383 052122 005015 045522 030466 OPR001: .ASCIZ <15><12>/RK611 VECTOR ADDRESS ( /
8384 052130 020061 042526 052103
8385 052136 051117 040440 042104
8386 052144 042522 051523 024040
8387 052152 000040
8388 052154 024440 036440 000040 OPR002: .ASCIZ / ) = /
8389 052162 045522 030466 020061 OPR003: .ASCIZ /RK611 VECTOR ADDRESS ( /
8390 052170 042526 052103 051117
8391 052176 040440 042104 042522
8392 052204 051523 024040 000040
8393 052212 045522 030466 020061 OPR004: .ASCIZ /RK611 PRIORITY ( /
8394 052220 051120 047511 044522
8395 052226 054524 024040 000040
8396 052234 005015 047062 020104 OPR006: .ASCIZ <15><12>/2ND PASS RUN TIME IS APPROX 13 MINUTES/<15><12>
8397 052242 040520 051523 051040
8398 052250 047125 052040 046511
8399 052256 020105 051511 040440
8400 052264 050120 047522 020130
8401 052272 031461 046440 047111
8402 052300 052125 051505 005015
8403 052306 000
8404 052307 015 025012 025052 OPR007: .ASCIZ <15><12>/***** PROGRAM HALTED *****/<15><12>
8405 052314 025052 020040 050040
8406 052322 047522 051107 046501
8407 052330 044040 046101 042524
8408 052336 020104 020040 025052
8409 052344 025052 025052 005015
8410 052352 000
8411 052353 040 000040 SPACE2: .ASCIZ / /
8412 052356 005015 051120 043517 ABORT: .ASCIZ <15><12>/PROGRAM ABORTED BECAUSE ERROR THRESHOLD EXCEEDED/<15><12>
8413 052364 040522 020115 041101
8414 052372 051117 042524 020104
8415 052400 042502 040503 051525
8416 052406 020105 051105 047522
8417 052414 020122 044124 042522
8418 052422 044123 046117 020104
8419 052430 054105 042503 042105
8420 052436 042105 005015 000
8421 052443 015 052012 051505 TSTBY1: .ASCIZ <15><12>/TEST /
8422 052450 020124 000
8423 052453 040 054502 040520 TSTBY2: .ASCIZ / BYPASSED/<15><12>
8424 052460 051523 042105 005015
8425 052466 000
```

```

8426          .SBTTL DATA HEADERS
8427
8428 052467      124 051505 020124 DH000A: .ASCIZ /TEST ERROR/
8429 052474 020040 042440 051122
8430 052502 051117      000
8431 052505      116 046525 020040 DH000B: .ASCIZ /NUM PC/
8432 052512 020040 050040 000103
8433 052520 042524 052123 020040 DH000C: .ASCII /TEST *RAP/<15><12>
8434 052526 020040 051124 050101
8435 052534 005015
8436 052536 052516 020115 020040      .ASCIZ /NUM PC/
8437 052544 020040 041520      000
8438 052551      105 050130 041505 DH001A: .ASCIZ /EXPECT ACU*AL EXPECT ACTUAL EXPECT ACTUAL/
8439 052556 020124 040440 052503
8440 052564 040524 020114 042440
8441 052572 050130 041505 020124
8442 052600 040440 052103 040525
8443 052606 020114 042440 050130
8444 052614 041505 020124 040440
8445 052622 052103 040525 000114
8446 052630 045522 051503 020061 DH001B: .ASCIZ /RKCS1 RKCS1 MESS A MESS A MESS B MESS B/
8447 052636 020040 045522 051503
8448 052644 020061 020040 042515
8449 052652 051523 040440 020040
8450 052660 042515 051523 040440
8451 052666 020040 042515 051523
8452 052674 041040 020040 042515
8453 052702 051523 041040      000
8454 052707      105 050130 041505 DH015A: .ASCIZ /EXPECT ACTUAL/
8455 052714 020124 040440 052103
8456 052722 040525 000114
8457 052726 045522 051503 020061 DH015B: .ASCIZ /RKCS1 RKCS1/
8458 052734 020040 045522 051503
8459 052742 000061
8460 052744 054105 042520 052103 DH017A: .ASCIZ /EXPECT ACTUAL PRESENT PREVIOUS BIT/
8461 052752 020040 041501 052524
8462 052760 046101 020040 051120
8463 052766 051505 047105 020124
8464 052774 051120 053105 052517
8465 053002 020123 044502 000124
8466 053010 045522 051115 020061 DH017B: .ASCIZ /RKMR1 RKMR1 BIT BIT COUNT/
8467 053016 020040 045522 051115
8468 053024 020061 020040 044502
8469 053032 020124 020040 020040
8470 053040 044502 020124 020040
8471 053046 020040 047503 047125
8472 053054 000124
8473 053056 054105 042520 052103 DH022A: .ASCIZ /EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL/
8474 053064 020040 041501 052524
8475 053072 046101 020040 054105
8476 053100 042520 052103 020040
8477 053106 041501 052524 046101
8478 053114 020040 054105 042520
8479 053122 052103 020040 041501
8480 053130 052524 046101      000
8481 053135      122 041513 030523 DH022B: .ASCIZ /RKCS1 RKCS1 RKBA RKBA RKWC RKWC/
  
```









```
8692 .SBTTL ERROR MESSAGES
8693
8694 055340 047125 054105 042520 EM000: .ASCIZ /UNEXPECTED MEMORY PARITY ENABLE TRAP/
8695 055346 052103 042105 046440
8696 055354 046505 051117 020131
8697 055362 040520 044522 054524
8698 055370 042440 040516 046102
8699 055376 020105 051124 050101
8700 055404 000
8701 055405 101 052124 046505 EM200: .ASCIZ /ATTEMPTING TO CHECK SEEK MESSAGE FROM READ HEADER/
8702 055412 052120 047111 020107
8703 055420 047524 041440 042510
8704 055426 045503 051440 042505
8705 055434 020113 042515 051523
8706 055442 043501 020105 051106
8707 055450 046517 051040 040505
8708 055456 020104 042510 042101
8709 055464 051105 000
8710 055467 101 052124 046505 EM201: .ASCIZ /ATTEMPTING TO CHECK SEEK MESSAGE FROM WRITE HEADER/
8711 055474 052120 047111 020107
8712 055502 047524 041440 042510
8713 055510 045503 051440 042505
8714 055516 020113 042515 051523
8715 055524 043501 020105 051106
8716 055532 046517 053440 044522
8717 055540 042524 044040 040505
8718 055546 042504 000122
8719 055552 052101 042524 050115 EM202: .ASCIZ /ATTEMPTING TO CHECK CLEAR DRIVE MESSAGE FROM READ HEADER/
8720 055560 044524 043516 052040
8721 055566 020117 044103 041505
8722 055574 020113 046103 040505
8723 055602 020122 051104 053111
8724 055610 020105 042515 051523
8725 055616 043501 020105 051106
8726 055624 046517 051040 040505
8727 055632 020104 042510 042101
8728 055640 051105 000
8729 055643 101 052124 046505 EM203: .ASCIZ /ATTEMPTING TO CHECK CLEAR DRIVE MESSAGE FROM WRITE HEADER/
8730 055650 052120 047111 020107
8731 055656 047524 041440 042510
8732 055664 045503 041440 042514
8733 055672 051101 042040 044522
8734 055700 042526 046440 051505
8735 055706 040523 042507 043040
8736 055714 047522 020115 051127
8737 055722 052111 020105 042510
8738 055730 042101 051105 000
8739 055735 101 052124 046505 EM204: .ASCIZ /ATTEMPTING A READ HEADER TO CHECK SECTOR PULSE DETECT/
8740 055742 052120 047111 020107
8741 055750 020101 042522 042101
8742 055756 044040 040505 042504
8743 055764 020122 047524 041440
8744 055772 042510 045503 051440
8745 056000 041505 047524 020122
8746 056006 052520 051514 020105
8747 056014 042504 042524 052103
```



8748	056022	000			
8749	056023	101	052124	046505	EM205: .ASCIZ /ATTEMPTING A WRITE HEADER TO CHECK INDEX PULSE DETECT/
8750	056030	052120	047111	020107	
8751	056036	020101	051127	052111	
8752	056044	020105	042510	042101	
8753	056052	051105	052040	020117	
8754	056060	044103	041505	020113	
8755	056066	047111	042504	020130	
8756	056074	052520	051514	020105	
8757	056102	042504	042524	052103	
8758	056110	000			
8759	056111	101	052124	046505	EM206: .ASCIZ /ATTEMPTING AN NPR READ OF ONE WORD/
8760	056116	052120	047111	020107	
8761	056124	047101	047040	051120	
8762	056132	051040	040505	020104	
8763	056140	043117	047440	042516	
8764	056146	053440	051117	000104	
8765	056154	052101	042524	050115	EM207: .ASCIZ /ATTEMPTING AN NPR READ/
8766	056162	044524	043516	040440	
8767	056170	020116	050116	020122	
8768	056176	042522	042101	000	
8769	056203	101	052124	046505	EM208: .ASCIZ /ATTEMPTING NPR READ (CHECKING ZERO DETECT/
8770	056210	052120	047111	020107	
8771	056216	050116	020122	042522	
8772	056224	042101	041440	042510	
8773	056232	045503	047111	020107	
8774	056240	042532	047522	042040	
8775	056246	052105	041505	000124	
8776	056254	052101	042524	050115	EM209: .ASCIZ /ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT INHIBIT/
8777	056262	044524	043516	047040	
8778	056270	051120	051040	040505	
8779	056276	020104	044527	044124	
8780	056304	041040	051525	040440	
8781	056312	042104	042522	051523	
8782	056320	044440	041516	042522	
8783	056326	042515	052116	044440	
8784	056334	044116	041111	052111	
8785	056342	000			
8786	056343	101	052124	046505	EM210: .ASCII /ATTEMPTING NPR READ WITH BUS ADDRESS INCREMENT INHIBIT/
8787	056350	052120	047111	020107	
8788	056356	050116	020122	042522	
8789	056364	042101	053440	052111	
8790	056372	020110	052502	020123	
8791	056400	042101	051104	051505	
8792	056406	020123	047111	051103	
8793	056414	046505	047105	020124	
8794	056422	047111	044510	044502	
8795	056430	124			
8796	056431	040	044103	041505	.ASCIZ / CHECKING ZERO DETECT/
8797	056436	044513	043516	055040	
8798	056444	051105	020117	042504	
8799	056452	042524	052103	000	
8800	056457	101	052124	046505	EM211: .ASCIZ /ATTEMPTING TO FORCE NON-EXISTENT MEMORY/
8801	056464	052120	047111	020107	
8802	056472	047524	043040	051117	
8803	056500	042503	047040	047117	

8804	056506	042455	044530	052123	
8805	056514	047105	020124	042515	
8806	056522	047515	054522	000	
8807	056527	101	052124	046505	EM212: .ASCIZ /ATTEMPTING TO CLEAR NON-EXISTENT MEMORY/
8808	056534	052120	047111	020107	
8809	056542	047524	041440	042514	
8810	056550	051101	047040	047117	
8811	056556	042455	044530	052123	
8812	056564	047105	020124	042515	
8813	056572	047515	054522	000	
8814	056577	124	051505	044524	EM213: .ASCIZ /TESTING EXTENDED MEMORY ADDRESSING BITS/
8815	056604	043516	042440	052130	
8816	056612	047105	042504	020104	
8817	056620	042515	047515	054522	
8818	056626	040440	042104	042522	
8819	056634	051523	047111	020107	
8820	056642	04 502	051524	000	
8821	056647	101	052124	046505	EM214: .ASCIZ /ATTEMPTING TO FORCE UNIBUS PARITY ERROR/
8822	056654	052120	047111	020107	
8823	056662	047524	043040	051117	
8824	056670	042503	052440	044516	
8825	056676	052502	020123	040520	
8826	056704	044522	054524	042440	
8827	056712	051122	051117	000	
8828	056717	101	052124	046505	EM215: .ASCIZ /ATTEMPTING NPR READ OF LOCATION PRIOR TO BAD PARITY/
8829	056724	052120	047111	020107	
8830	056732	050116	020122	042522	
8831	056740	042101	047440	020106	
8832	056746	047514	040503	044524	
8833	056754	047117	050040	044522	
8834	056762	051117	052040	020117	
8835	056770	040502	020104	040520	
8836	056776	044522	054524	000	
8837	057003	101	052124	046505	EM216: .ASCIZ /ATTEMPTING TO CLEAR UNIBUS PARITY ERROR/
8838	057010	052120	047111	020107	
8839	057016	047524	041440	042514	
8840	057024	051101	052440	044516	
8841	057032	052502	020123	040520	
8842	057040	044522	054524	042440	
8843	057046	051122	051117	000	
8844	057053	101	052124	046505	EM217: .ASCIZ /ATTEMPTING 18 BIT NPR READ/
8845	057060	052120	047111	020107	
8846	057066	034061	041040	052111	
8847	057074	047040	051120	051040	
8848	057102	040505	000104		
8849	057106	052101	042524	050115	EM218: .ASCIZ /ATTEMPTING 18 BIT NPR READ CHECKING ZERO DETECT/
8850	057114	044524	043516	030440	
8851	057122	020070	044502	020124	
8852	057130	050116	020122	042522	
8853	057136	042101	041440	042510	
8854	057144	045503	047111	020107	
8855	057152	042532	047522	042040	
8856	057160	052105	041505	000124	
8857	057166	052101	042524	050115	EM219: .ASCIZ /ATTEMPTING 18 BIT NPR READ WITH BIT 16 (PA) SET/
8858	057174	044524	043516	030440	
8859	057202	020070	044502	020124	

8860	057210	050116	020122	042522	
8861	057216	042101	053440	052111	
8862	057224	020110	044502	020124	
8863	057232	033061	024040	040520	
8864	057240	020051	042523	000124	
8865	057246	052101	042524	050115	EM220: .ASCII /ATTEMPTING 18 BIT NPR READ WITH BIT 16 (PA) SET/
8866	057254	044524	043516	030440	
8867	057262	020070	044502	020124	
8868	057270	050116	020122	042522	
8869	057276	042101	053440	052111	
8870	057304	020110	044502	020124	
8871	057312	033061	024040	040520	
8872	057320	020051	042523	124	
8873	057325	101	052124	046505	EM221: .ASCIZ /ATTEMPTING SIMULATION OF DATA IN READ HEADER/
8874	057332	052120	047111	020107	
8875	057340	044523	052515	040514	
8876	057346	044524	047117	047440	
8877	057354	020106	040504	040524	
8878	057362	044440	020116	042522	
8879	057370	042101	044040	040505	
8880	057376	042504	000122		
8881	057402	052101	042524	050115	EM222: .ASCIZ /ATTEMPTING READ HEADER IN MAINTANENCE MODE/
8882	057410	044524	043516	051040	
8883	057416	040505	020104	042510	
8884	057424	042101	051105	044440	
8885	057432	020116	040515	047111	
8886	057440	040524	042516	041516	
8887	057446	020105	047515	042504	
8888	057454	000			
8889	057455	101	052124	046505	EM223: .ASCIZ /ATTEMPTING DATA BUFFER READ AFTER READ HEADER/
8890	057462	052120	047111	020107	
8891	057470	040504	040524	041040	
8892	057476	043125	042506	020122	
8893	057504	042522	042101	040440	
8894	057512	052106	051105	051040	
8895	057520	040505	020104	042510	
8896	057526	042101	051105	000	
8897	057533	101	052124	046505	EM224: .ASCIZ /ATTEMPTING SIMULATION OF DATA IN READ HEADER (18 BIT FORMAT)/
8898	057540	052120	047111	020107	
8899	057546	044523	052515	040514	
8900	057554	044524	047117	047440	
8901	057562	020106	040504	040524	
8902	057570	044440	020116	042522	
8903	057576	042101	044040	040505	
8904	057604	042504	020122	030450	
8905	057612	020070	044502	020124	
8906	057620	047506	046522	052101	
8907	057626	000051			
8908	057630	052101	042524	050115	EM225: .ASCIZ /ATTEMPTING READ HEADER (18 BIT FORMAT) IN MAINT MODE/
8909	057636	044524	043516	051040	
8910	057644	040505	020104	042510	
8911	057652	042101	051105	024040	
8912	057660	034061	041040	052111	
8913	057666	043040	051117	040515	
8914	057674	024524	044440	020116	
8915	057702	040515	047111	020124	

8916	057710	047515	042504	000	
8917	057715	101	052124	046505	EM226: .ASCII /ATTEMPTING DATA BUFFER READ AFTER <12><15>
8918	057722	052120	047111	020107	
8919	057730	040504	040524	041040	
8920	057736	043125	042506	020122	
8921	057744	042522	042101	040440	
8922	057752	052106	051105	006412	
8923	057760	042522	042101	044040	.ASCIZ /READ HEADER IN 18 BIT FORMAT/
8924	057766	040505	042504	020122	
8925	057774	047111	030440	020070	
8926	060002	044502	020124	047506	
8927	060010	046522	052101	000	
8928	060015	101	052124	046505	EM227: .ASCII /ATTEMPTING TO CHECK SYNCH DETECT ON READ HEADER/<15><12>
8929	060022	052120	047111	020107	
8930	060030	047524	041440	042510	
8931	060036	045503	051440	047131	
8932	060044	044103	042040	052105	
8933	060052	041505	020124	047117	
8934	060060	051040	040505	020104	
8935	060066	042510	042101	051105	
8936	060074	005015			
8937	060076	044103	041505	044513	.ASCIZ /CHECKING ZERO DETECT/
8938	060104	043516	055040	051105	
8939	060112	020117	042504	042524	
8940	060120	052103	000		
8941	060123	101	052124	046505	EM230: .ASCII /ATTEMPTING TO CHECK WRITING OF ZEROES BETWEEN INDEX/<15><12>
8942	060130	052120	047111	020107	
8943	060136	047524	041440	042510	
8944	060144	045503	053440	044522	
8945	060152	044524	043516	047440	
8946	060160	020106	042532	047522	
8947	060166	051505	041040	052105	
8948	060174	042527	047105	044440	
8949	060202	042116	054105	005015	
8950	060210	047101	020104	042523	.ASCIZ /AND SECTOR PULSE/
8951	060216	052103	051117	050040	
8952	060224	046125	042523	000	
8953	060231	101	052124	046505	EM231: .ASCII /ATTEMPTING TO RESET WRITE GATE BY SETTING/<15><12>
8954	060236	052120	047111	020107	
8955	060244	047524	051040	051505	
8956	060252	052105	053440	044522	
8957	060260	042524	043440	052101	
8958	060266	020105	054502	051440	
8959	060274	052105	044524	043516	
8960	060302	005015			
8961	060304	042523	052103	051117	.ASCIZ /SECTOR PULSE IN A WRITE HEADER COMMAND/
8962	060312	050040	046125	042523	
8963	060320	044440	020116	020101	
8964	060326	051127	052111	020105	
8965	060334	042510	042101	051105	
8966	060342	041440	046517	040515	
8967	060350	042116	000		
8968	060353	101	052124	046505	EM232: .ASCII /ATTEMPTING TO SET WRITE GATE BY RESETTING/<15><12>
8969	060360	052120	047111	020107	
8970	060366	047524	051440	052105	
8971	060374	053440	044522	042524	

8972	060402	043440	052101	020105	
8973	060410	054502	051040	051505	
8974	060416	052105	044524	043516	
8975	060424	005015			
8976	060426	042523	052103	051117	.ASCIZ /SECTOR PULSE IN A WRITE HEADER COMMAND/
8977	060434	050040	046125	042523	
8978	060442	044440	020116	020101	
8979	060450	051127	052111	020105	
8980	060456	042510	042101	051105	
8981	060464	041440	046517	040515	
8982	060472	042116	000		
8983	060475	101	052124	046505	EM233: .ASCIZ /ATTEMPTING TO WRITE SYNCH OF HEADER/
8984	060502	052120	047111	020107	
8985	060510	047524	053440	044522	
8986	060516	042524	051440	047131	
8987	060524	044103	047440	020106	
8988	060532	042510	042101	051105	
8989	060540	000			
8990	060541	101	052124	046505	EM234: .ASCIZ /ATTEMPTING TO WRITE HEADER DATA/
8991	060546	052120	047111	020107	
8992	060554	047524	053440	044522	
8993	060562	042524	044040	040505	
8994	060570	042504	020122	040504	
8995	060576	040524	000		
8996	060601	101	052124	046505	EM235: .ASCII /ATTEMPTING TO RESET WRITE GATE WITH SECOND/<15><12>
8997	060606	052120	047111	020107	
8998	060614	047524	051040	051505	
8999	060622	052105	053440	044522	
9000	060630	042524	043440	052101	
9001	060636	020105	044527	044124	
9002	060644	051440	041505	047117	
9003	060652	006504	012		
9004	060655	111	042116	054105	.ASCIZ /INDEX PULSE OF WRITE HEADER/
9005	060662	050040	046125	042523	
9006	060670	047440	020106	051127	
9007	060676	052111	020105	042510	
9008	060704	042101	051105	000	
9009	060711	101	052124	046505	EM236: .ASCIZ /ATTEMPTING TO COMPLETE WRITE HEADER IN MAINT MODE/
9010	060716	052120	047111	020107	
9011	060724	047524	041440	046517	
9012	060732	046120	052105	020105	
9013	060740	051127	052111	020105	
9014	060746	042510	042101	051105	
9015	060754	044440	020116	040515	
9016	060762	047111	020124	047515	
9017	060770	042504	000		
9018	060773	101	052124	046505	EM237: .ASCIZ /ATTEMPTING TO WRITE GAP OR DATA SYNCH/
9019	061000	052120	047111	020107	
9020	061006	047524	053440	044522	
9021	061014	042524	043440	050101	
9022	061022	047440	020122	040504	
9023	061030	040524	051440	047131	
9024	061036	044103	000		
9025	061041	101	052124	046505	EM238: .ASCIZ /ATTEMPTING TO WRITE DATA FIELD/
9026	061046	052120	047111	020107	
9027	061054	047524	053440	044522	

9028	061062	042524	042040	052101	
9029	061070	020101	044506	046105	
9030	061076	000104			
9031	061100	052101	042524	050115	EM239: .ASCIZ /ATTEMPTING TO WRITE SYNCH OF HEADER USING 24 SECTOR FORMAT/
9032	061106	047111	020107	047524	
9033	061114	053440	044522	042524	
9034	061122	051440	047131	044103	
9035	061130	047440	020106	042510	
9036	061136	042101	051105	052440	
9037	061144	044523	043516	031040	
9038	061152	020064	042523	052103	
9039	061160	051117	043040	051117	
9040	061166	040515	000124		
9041	061172	052101	042524	050115	EM240: .ASCIZ /ATTEMPTING TO WRITE HEADER DATA USING 24 SECTOR FORMAT/
9042	061200	044524	043516	052040	
9043	061206	020117	051127	052111	
9044	061214	020105	042510	042101	
9045	061222	051105	042040	052101	
9046	061230	020101	051525	047111	
9047	061236	020107	032062	051440	
9048	061244	041505	047524	020122	
9049	061252	047506	046522	052101	
9050	061260	000			
9051	061261	101	052124	046505	EM241: .ASCIZ /ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 26 SECTOR)/
9052	061266	052120	047111	020107	
9053	061274	047524	043040	051117	
9054	061302	042503	043040	051117	
9055	061310	040515	020124	051105	
9056	061316	047522	020122	041450	
9057	061324	046506	020124	020075	
9058	061332	033062	051440	041505	
9059	061340	047524	024522	000	
9060	061345	101	052124	046505	EM242: .ASCIZ /ATTEMPTING TO FORCE FORMAT ERROR (CFMT = 24 SECTOR)/
9061	061352	052120	047111	020107	
9062	061360	047524	043040	051117	
9063	061366	042503	043040	051117	
9064	061374	040515	020124	051105	
9065	061402	047522	020122	041450	
9066	061410	046506	020124	020075	
9067	061416	032062	051440	041505	
9068	061424	047524	024522	000	
9069	061431	101	052124	046505	EM243: .ASCIZ /ATTEMPTING TO FORCE CONTROLLER ERROR WITH FAULT BIT IN DRIVE MESS/
9070	061436	044520	043516	052040	
9071	061444	020117	047506	041522	
9072	061452	020105	047503	052116	
9073	061460	047522	046114	051105	
9074	061466	042440	051122	051117	
9075	061474	053440	052111	020110	
9076	061502	040506	046125	020124	
9077	061510	044502	020124	047111	
9078	061516	042040	044522	042526	
9079	061524	046440	051505	000123	
9080	061532	052101	042524	050115	EM244: .ASCIZ /ATTEMPTING TO CLEAR ERROR/
9081	061540	044524	043516	052040	
9082	061546	020117	046103	040505	
9083	061554	020122	051105	047522	

9084	061562	000122				
9085	061564	047503	046515	047101	EM3000: .ASCIZ	/COMMAND AND STATUS REG 1 INCORRECT/
9086	061572	020104	047101	020104		
9087	061600	052123	052101	051525		
9088	061606	051040	043505	030440		
9089	061614	044440	041516	051117		
9090	061622	042522	052103	000		
9091	061627	115	051505	040523	EM3001: .ASCIZ	/MESSAGE A INCORRECT/
9092	061634	042507	040440	044440		
9093	061642	041516	051117	042522		
9094	061650	052103	000			
9095	061653	115	051505	040523	EM3002: .ASCIZ	/MESSAGE B INCORRECT/
9096	061660	042507	041040	044440		
9097	061666	041516	051117	042522		
9098	061674	052103	000			
9099	061677	103	030523	044440	EM3003: .ASCIZ	/CS1 INCORRECT AFTER SENDING DRIVE CLEAR/
9100	061704	041516	051117	042522		
9101	061712	052103	040440	052106		
9102	061720	051105	051440	047105		
9103	061726	044504	043516	042040		
9104	061734	044522	042526	041440		
9105	061742	042514	051101	000		
9106	061747	103	030523	044440	EM3004: .ASCIZ	/CS1 INCORRECT AFTER AFTER DATA SIMULATION/
9107	061754	041516	051117	042522		
9108	061762	052103	040440	052106		
9109	061770	051105	040440	052106		
9110	061776	051105	042040	052101		
9111	062004	020101	044523	052515		
9112	062012	040514	044524	047117		
9113	062020	000				
9114	062021	115	044501	052116	EM3005: .ASCII	/MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9115	062026	051040	043505	020056		
9116	062034	020061	047111	047503		
9117	062042	051122	041505	020124		
9118	062050	052504	044522	043516		
9119	062056	042040	052101	020101		
9120	062064	044523	052515	040514		
9121	062072	044524	047117			
9122	062076	005015	043101	042524	.ASCIZ	<15><12>/AFTER SECTOR PULSE/
9123	062104	020122	042523	052103		
9124	062112	051117	050040	046125		
9125	062120	042523	000			
9126	062123	115	044501	052116	EM3006: .ASCII	/MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9127	062130	051040	043505	020056		
9128	062136	020061	047111	047503		
9129	062144	051122	041505	020124		
9130	062152	052504	044522	043516		
9131	062160	042040	052101	020101		
9132	062166	044523	052515	040514		
9133	062174	044524	047117			
9134	062200	005015	043101	042524	.ASCIZ	<15><12>/AFTER INDEX PULSE/
9135	062206	020122	047111	042504		
9136	062214	020130	052520	051514		
9137	062222	000105				
9138	062224	040515	047111	020124	EM3007: .ASCII	/MAINT REG. 1 INCORRECT DURING DATA SIMULATION/
9139	062232	042522	027107	030440		

9140	062240	044440	041516	051117	
9141	062246	042522	052103	042040	
9142	062254	051125	047111	020107	
9143	062262	040504	040524	051440	
9144	062270	046511	046125	052101	
9145	062276	047511	116		
9146	062301	015	047012	020117	.ASCIZ <15><12>,NO SECTOR OR INDEX PULSE SUPPLIED/
9147	062306	042523	052103	051117	
9148	062314	047440	020122	047111	
9149	062322	042504	020130	052520	
9150	062330	051514	020105	052523	
9151	062336	050120	044514	042105	
9152	062344	000			
9153	062345	102	051525	040440	EM3008: .ASCIZ /BUS ADDRESS INCORRECT AFTER SENDING DRIVE CLEAR/
9154	062352	042104	042522	051523	
9155	062360	044440	041516	051117	
9156	062366	042522	052103	040440	
9157	062374	052106	051105	051440	
9158	062402	047105	044504	043516	
9159	062410	042040	044522	042526	
9160	062416	041440	042514	051101	
9161	062424	000			
9162	062425	127	051117	020104	EM3009: .ASCIZ /WORD COUNT INCORRECT AFTER SENDING DRIVE CLEAR/
9163	062432	047503	047125	020124	
9164	062440	047111	047503	051122	
9165	062446	041505	020124	043101	
9166	062454	042524	020122	042523	
9167	062462	042116	047111	020107	
9168	062470	051104	053111	020105	
9169	062476	046103	040505	000122	
9170	062504	051503	020061	044103	EM3010: .ASCIZ /CS1 CHANGED DURING COMMAND EXECUTION/
9171	062512	047101	042507	020104	
9172	062520	052504	044522	043516	
9173	062526	041440	046517	040515	
9174	062534	042116	042440	042530	
9175	062542	052503	044524	047117	
9176	062550	000			
9177	062551	102	051525	040440	EM3011: .ASCIZ /BUS ADDRESS CHANGED BEFORE INDEX PULSE/
9178	062556	042104	042522	051523	
9179	062564	041440	040510	043516	
9180	062572	042105	041040	043105	
9181	062600	051117	020105	047111	
9182	062606	042504	020130	052520	
9183	062614	051514	000105		
9184	062620	047527	042122	041440	EM3012: .ASCIZ /WORD COUNT CHANGED BEFORE INDEX PULSE/
9185	062626	052517	052116	041440	
9186	062634	040510	043516	042105	
9187	062642	041040	043105	051117	
9188	062650	020105	047111	042504	
9189	062656	020130	052520	051514	
9190	062664	000105			
9191	062666	051503	020061	044103	EM3013: .ASCIZ /CS1 CHANGE AFTER INDEX PULSE/
9192	062674	047101	042507	040440	
9193	062702	052106	051105	044440	
9194	062710	042116	054105	050040	
9195	062716	046125	042523	000	



9196	062723	115	044501	052116	EM3014: .ASCIZ /MAINT REG 1 INCORRECT BEFORE INDEX PULSE/
9197	062730	051040	043505	030440	
9198	062736	044440	041516	051117	
9199	062744	042522	052103	041040	
9200	062752	043105	051117	020105	
9201	062760	047111	042504	020130	
9202	062766	052520	051514	000105	
9203	062774	052502	020123	042101	EM3015: .ASCIZ /BUS ADDRESS CHANGED AFTER INDEX PULSE/
9204	063002	051104	051505	020123	
9205	063010	044103	047101	042507	
9206	063016	020104	043101	042524	
9207	063024	020122	047111	042504	
9208	063032	020130	052520	051514	
9209	063040	000105			
9210	063042	047527	042122	041440	EM3016: .ASCIZ /WORD COUNT CHANGED AFTER INDEX PULSE/
9211	063050	052517	052116	041440	
9212	063056	040510	043516	042105	
9213	063064	040440	052106	051105	
9214	063072	044440	042116	054105	
9215	063100	050040	046125	042523	
9216	063106	000			
9217	063107	103	046517	040515	EM3018: .ASCIZ /COMMAND AND STATUS REG. 2 INCORRECT/
9218	063114	042116	040440	042116	
9219	063122	051440	040524	052524	
9220	063130	020123	042522	027107	
9221	063136	031040	044440	041516	
9222	063144	051117	042522	052103	
9223	063152	000			
9224	063153	102	051525	040440	EM3019: .ASCIZ /BUS ADD REG INCORRECT/
9225	063160	042104	051040	043505	
9226	063166	044440	041516	051117	
9227	063174	042522	052103	000	
9228	063201	127	051117	020104	EM3020: .ASCIZ /WORD COUNT REG INCORRECT/
9229	063206	047503	047125	020124	
9230	063214	042522	020107	047111	
9231	063222	047503	051122	041505	
9232	063230	000124			
9233	063232	040504	040524	051040	EM3021: .ASCIZ /DATA READ INCORRECT/
9234	063240	040505	020104	047111	
9235	063246	047503	051122	041505	
9236	063254	000124			
9237	063256	051503	020061	047111	EM3022: .ASCIZ /CS1 INCORRECT AFTER READING DATA BUFFER/
9238	063264	047503	051122	041505	
9239	063272	020124	043101	042524	
9240	063300	020122	042522	042101	
9241	063306	047111	020107	040504	
9242	063314	040524	041040	043125	
9243	063322	042506	000122		
9244	063326	051503	020062	047111	EM3023: .ASCIZ /CS2 INCORRECT AFTER READING DATA BUFFER/
9245	063334	047503	051122	041505	
9246	063342	020124	043101	042524	
9247	063350	020122	042522	042101	
9248	063356	047111	020107	040504	
9249	063364	040524	041040	043125	
9250	063372	042506	000122		
9251	063376	051105	047522	020122	EM3024: .ASCIZ /ERROR REG INCORRECT/

9252	063404	042522	020107	047111	
9253	063412	047503	051122	041505	
9254	063420	000124			
9255	063422	040515	047111	020124	EM3025: .ASCIZ /MAINT REG 1 INCORRECT AFTER INDEX PULSE/
9256	063430	042522	020107	020061	
9257	063436	047111	047503	051122	
9258	063444	041505	020124	043101	
9259	063452	042524	020122	047111	
9260	063460	042504	020130	052520	
9261	063466	051514	000105		
9262	063472	051503	020061	047111	EM3026: .ASCIZ /CS1 INCORRECT AFTER COMMAND COMPLETION/
9263	063500	047503	051122	041505	
9264	063506	020124	043101	042524	
9265	063514	020122	047503	046515	
9266	063522	047101	020104	047503	
9267	063530	050115	042514	044524	
9268	063536	047117	000		
9269	063541	103	031123	044440	EM3027: .ASCIZ /CS2 INCORRECT AFTER COMMAND COMPLETION/
9270	063546	041516	051117	042522	
9271	063554	052103	040440	052106	
9272	063562	051105	041440	046517	
9273	063570	040515	042116	041440	
9274	063576	046517	046120	052105	
9275	063604	047511	000116		
9276	063610	051503	020061	047111	EM3028: .ASCIZ /CS1 INCORRECT AFTER READING EMPTY SILO/
9277	063616	047503	051122	041505	
9278	063624	020124	043101	042524	
9279	063632	020122	042522	042101	
9280	063640	047111	020107	046505	
9281	063646	052120	020131	044523	
9282	063654	047514	000		
9283	063657	103	031123	044440	EM3029: .ASCIZ /CS2 INCORRECT AFTER READING EMPTY SILO/
9284	063664	041516	051117	042522	
9285	063672	052103	040440	052106	
9286	063700	051105	051040	040505	
9287	063706	044504	043516	042440	
9288	063714	050115	054524	051440	
9289	063722	046111	000117		
9290	063726	040515	047111	020124	EM3030: .ASCIZ /MAINT REG 1 INCORRECT/
9291	063734	042522	020107	020061	
9292	063742	047111	047503	051122	
9293	063750	041505	000124		
9294	063754	051104	053111	020105	EM3031: .ASCIZ /DRIVE STATUS REG INCORRECT/
9295	063762	052123	052101	051525	
9296	063770	051040	043505	044440	
9297	063776	041516	051117	042522	
9298	064004	052103	000		
9299	064007	105	051122	051117	EM3032: .ASCIZ /ERROR REG INCORRECT/
9300	064014	051040	043505	044440	
9301	064022	041516	051117	042522	
9302	064030	052103	000		
9303	064033	115	030522	044440	EMW1. .ASCIZ /MR1 INCORRECT ON 1ST UPWARD TRANSITION OF MAINT CLOCK/
9304	064040	041516	051117	042522	
9305	064046	052103	047440	020116	
9306	064054	051461	020124	050125	
9307	064062	040527	042122	052040	

9308	064070	040522	051516	051511	
9309	064076	044524	047117	047440	
9310	064104	020106	040515	047111	
9311	064112	020124	046103	041517	
9312	064120	000113			
9313	064122	051115	020061	047111	EMW2: .ASCIZ /MR1 INCORRECT ON 1ST DOWNWARD TRANSITION OF MAINT CLOCK/
9314	064130	047503	051122	041505	
9315	064136	020124	047117	030440	
9316	064144	052123	042040	053517	
9317	064152	053516	051101	020104	
9318	064160	051124	047101	044523	
9319	064166	052123	047511	020116	
9320	064174	043117	046440	044501	
9321	064202	052116	041440	047514	
9322	064210	045503	000		
9323	064213	115	030522	044440	EMW3: .ASCIZ /MR1 INCORRECT ON 2ND UPWARD TRANSITION OF MAINT CLOCK/
9324	064220	041516	051117	042522	
9325	064226	052103	047440	020116	
9326	064234	047062	020104	050125	
9327	064242	040527	042122	052040	
9328	064250	040522	051516	051511	
9329	064256	044524	047117	047440	
9330	064264	020106	040515	047111	
9331	064272	020124	046103	041517	
9332	064300	000113			
9333	064302	051115	020061	047111	EMW4: .ASCIZ /MR1 INCORRECT ON 2ND DOWNWARD TRANSITION OF MAINT CLOCK/
9334	064310	047503	051122	041505	
9335	064316	020124	047117	031040	
9336	064324	042116	042040	053517	
9337	064332	053516	051101	020104	
9338	064340	051124	047101	044523	
9339	064346	052123	047511	020116	
9340	064354	043117	046440	044501	
9341	064362	052116	041440	047514	
9342	064370	045503	000		

9343				.SBTTL DATA PATTERNS
9344				
9345		064374		.EVEN
9346	064374	000000		PATTERN: .WORD 000000
9347	064376	177777		.WORD 177777
9348	064400	125252		.WORD 125252
9349	064402	052515		.WORD 052515
9350	064404	101706		.WORD 101706
9351	064406	060422		.WORD 060422
9352	064410	130715		.WORD 130715
9353	064412	177777		HEAD1: .WORD 177777
9354	064414	000000		.WORD 000000
9355	064416	177777		.WORD 177777
9356	064420	000000		HEAD2: .WORD 000000
9357	064422	177777		.WORD 177777
9358	064424	000000		.WORD 000000
9359	064426	125252		HEAD3: .WORD 125252
9360	064430	052525		.WORD 052525
9361	064432	125252		.WORD 125252
9362	064434	052525		.WORD 052525
9363	064436	125252		.WORD 125252
9364	064440	052525		.WORD 052525
9365	064442	044444		HEAD4: .WORD 044444
9366	064444	022222		.WORD 022222
9367	064446	111111		.WORD 111111
9368	064450	052012		HEAD5: .WORD 052012
9369	064452	100520		.WORD 100520
9370	064454	052012		.WORD 052012
9371	064456	155555		HEAD6: .WORD 155555
9372	064460	066666		.WORD 066666
9373	064462	155555		.WORD 155555
9374	064464	104210		HEAD7: .WORD 104210
9375	064466	104210		.WORD 104210
9376	064470	104210		.WORD 104210
9377	064472	100	100	NPRBUF: .BYTE 100,100
9378	064474	101	101	.BYTE 101,101
9379	064476	102	102	.BYTE 102,102
9380	064500	103	103	.BYTE 103,103
9381	064502	104	104	.BYTE 104,104
9382	064504	105	105	.BYTE 105,105
9383	064506	106	106	.BYTE 106,106
9384	064510	107	107	.BYTE 107,107
9385	064512	110	110	.BYTE 110,110
9386	064514	111	111	.BYTE 111,111
9387	064516	112	112	.BYTE 112,112
9388	064520	113	113	.BYTE 113,113
9389	064522	114	114	.BYTE 114,114
9390	064524	115	115	.BYTE 115,115
9391	064526	116	116	.BYTE 116,116
9392	064530	117	117	.BYTE 117,117
9393	064532	120	120	.BYTE 120,120
9394	064534	121	121	.BYTE 121,121
9395	064536	122	122	.BYTE 122,122
9396	064540	123	123	.BYTE 123,123
9397	064542	124	124	.BYTE 124,124
9398	064544	125	125	.BYTE 125,125

9399	064546	126	126	.BYTE	126,126
9400	064550	127	127	.BYTE	127,127
9401	064552	130	130	.BYTE	130,130
9402	064554	131	131	.BYTE	131,131
9403	064556	132	132	.BYTE	132,132
9404	064560	133	133	.BYTE	133,133
9405	064562	134	134	.BYTE	134,134
9406	064564	135	135	.BYTE	135,135
9407	064566	136	136	.BYTE	136,136
9408	064570	137	137	.BYTE	137,137
9409	06457	140	140	.BYTE	140,140
9410	0645 4	141	141	.BYTE	141,141
9411	064576	142	142	.BYTE	142,142
9412	064600	143	143	.BYTE	143,143
9413	064602	144	144	.BYTE	144,144
9414	064604	145	145	.BYTE	145,145
9415	064606	146	146	.BYTE	146,146
9416	064610	147	147	.BYTE	147,147
9417	064612	150	150	.BYTE	150,150
9418	064614	151	151	.BYTE	151,151
9419	064616	152	152	.BYTE	152,152
9420	064620	153	153	.BYTE	153,153
9421	064622	154	154	.BYTE	154,154
9422	064624	155	155	.BYTE	155,155
9423	064626	156	156	.BYTE	156,156
9424	064630	157	157	.BYTE	157,157
9425	064632	160	160	.BYTE	160,160
9426	064634	161	161	.BYTE	161,161
9427	064636	162	162	.BYTE	162,162
9428	064640	163	163	.BYTE	163,163
9429	064642	164	164	.BYTE	164,164
9430	064644	165	165	.BYTE	165,165
9431	064646	166	166	.BYTE	166,166
9432	064650	167	167	.BYTE	167,167
9433	064652	170	170	.BYTE	170,170
9434	064654	171	171	.BYTE	171,171
9435	064656	172	172	.BYTE	172,172
9436	064660	173	173	.BYTE	173,173
9437	064662	174	174	.BYTE	174,174
9438	064664	175	175	.BYTE	175,175
9439	064666	176	176	.BYTE	176,176
9440	064670	177	177	.BYTE	177,177
9441	064672	200	200	.BYTE	200,200
9442	064674	201	201	.BYTE	201,201
9443	064676	202	202	.BYTE	202,202
9444	064700	203	203	.BYTE	203,203
9445	064702	204	204	.BYTE	204,204
9446	064704	205	205	.BYTE	205,205
9447	064706	206	206	.BYTE	206,206
9448	064710	207	207	.BYTE	207,207
9449	064712	210	210	.BYTE	210,210
9450	064714	000102		.BLKW	06.
9451		064730		BADPAR=	WRBUFF+14
9452		000001		.END	

ABASE = 177440	1009#	1247	1288			
ABORT = 052356	7416	8412#				
ACDW1 = 000000	1247	1290				
ACDW2 = 000000	1247	1291				
ACLO = 000010	1104#					
ACPUOP= 000000	1247	1262				
ADDW0 = 000000	1247					
ADDW1 = 000000	1247					
ADDW10= 000000	1247					
ADDW11= 000000	1247					
ADDW12= 000000	1247					
ADDW13= 000000	1247					
ADDW14= 000000	1247					
ADDW15= 000000	1247					
ADDW2 = 000000	1247					
ADDW3 = 000000	1247					
ADDW4 = 000000	1247					
ADDW5 = 000000	1247					
ADDW6 = 000000	1247					
ADDW7 = 000000	1247					
ADDW8 = 000000	1247					
ADDW9 = 000000	1247					
ADEVCT- 000000	1247	1253				
ADEVM - 000000	1247	1289				
AENV - 000000	1247	1258				
AENVM - 000000	1247	1259				
AFATAL= 000000	1247	1250				
AMADR1= 000000	1247	1275				
AMADR2= 000000	1247	1279				
AMADR3= 000000	1247	1282				
AMADR4= 000000	1247	1285				
AMAMS1= 000000	1247	1269				
AMAMS2= 000000	1247	1277				
AMAMS3= 000000	1247	1280				
AMAMS4= 000000	1247	1283				
AMSGAD= 000000	1247	1255				
AMSGLG= 000000	1247	1256				
AMSGTY= 000000	1247	1249				
AMTYP1- 000000	1247	1270				
AMTYP2= 000000	1247	1278				
AMTYP3= 000000	1247	1281				
AMTYP4= 000000	1247	1284				
APASS = 000000	1247	1252				
APRIOR= 000005	1008#	1247				
APTCSU= 000040	7293#	7471				
APTENV= 000001	7249	7291#	7325	7464		
APTSIZ= 000200	2282	7290#				
APTSPO= 000100	7251	7292#	7466			
ASWREG= 000000	1247	1260				
ATESTN= 000000	1247	1251				
AUNIT = 000000	1247	1254				
AUSWR = 000000	1247	1261				
AVECT1= 120210	1007#	1247	1286			
AVECT2= 000000	1247	1287				
BADPAR= 06473C	9451#					
BAI = 00002C	1067#	3509	3524	3554	3583	3596







CZR6CEO RK611 DSKLS CTRL PRT3  
CZR6CE.P11 27-AUG-81 10:24

MACY11 30(1046) 28-AUG-81 10:35 M 14  
PAGE 183  
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0181

DH072D 054346  
DH150A 054423  
DH150B 054457  
DH164A 054515  
DH164B 054561  
DH170A 054627  
DH170B 054726  
DH171A 055024  
DH171B 055043  
DH175A 055061  
DH175B 055160  
DH211A 055255  
DH211B 055303  
DI = 040000  
DISPLA 001142  
DISPRE 000174  
DLT = 100000  
DMD - 000040

8305	8600#												
8323	8608#												
8325	8613#												
8333	8619#												
8335	8626#												
8343	8633#												
8345	8644#												
8353	8655#												
8355	8658#												
8363	8373	8661#											
8365	8375	8672#											
8377	8683#												
8379	8687#												
1059#													
1220#	2270*	2278*	7182*	7312*									
1149#	2278												
1078#													
1120#	2387	2394	2395	2432	2439	2440	2475	2482	2483	2519	2526	2527	
2567	2568	2569	2572	2573	2584	2585	2586	2602	2656	2657	2658	2661	
2662	2673	2675	2676	2679	2681	2737	2738	2739	2742	2743	2755	2813	
2814	2815	2820	2821	2845	2848	2849	2850	2851	2879	2880	2881	2884	
2905	2906	2907	2908	2935	2937	2938	2941	2944	2945	2952	2996	3005	
3006	3010	3011	3012	3015	3017	3018	3085	3094	3095	3099	3100	3101	
3104	3107	3108	3137	3138	3201	3209	3210	3214	3215	3216	3219	3222	
3223	3257	3258	3330	3338	3339	3343	3344	3345	3348	3351	3352	3390	
3391	3428	3429	3504	3513	3514	3518	3519	3520	3523	3526	3527	3559	
3560	3622	3631	3632	3636	3637	3638	3641	3643	3644	3725	3730	3731	
3735	3736	3737	3740	3742	3743	3784	3792	3793	3797	3798	3799	3802	
3805	3806	3889	3893	3894	3898	3899	3900	3903	3905	3906	3947	3955	
3956	3960	3961	3962	3965	3968	3969	4051	4056	4057	4061	4062	4063	
4066	4068	4069	4110	4118	4119	4123	4124	4125	4128	4131	4132	4189	
4198	4199	4203	4204	4205	4208	4212	4213	4245	4246	4320	4323	4324	
4327	4328	4430	4433	4434	4437	4438	4540	4543	4544	4547	4548	4650	
4653	4654	4657	4658	4760	4763	4764	4767	4768	4869	4872	4873	4876	
4877	4972	4975	4976	4979	4980	5036	5039	5040	5043	5044	5045	5046	
5143	5144	5145	5151	5152	5156	5157	5158	5161	5162	5165	5166	5184	
5191	5219	5225	5226	5230	5231	5232	5235	5237	5238	5241	5242	5245	
5293	5295	5296	5299	5301	5306	5307	5339	5345	5346	5350	5351	5352	
5355	5357	5358	5361	5362	5365	5413	5415	5416	5419	5421	5426	5427	
5458	5464	5465	5469	5470	5471	5474	5476	5477	5480	5481	5484	5532	
5534	5535	5538	5540	5545	5546	5577	5583	5584	5588	5589	5590	5593	
5595	5596	5599	5600	5603	5651	5653	5654	5657	5659	5664	5665	5696	
5702	5703	5707	5708	5709	5712	5714	5715	5718	5719	5722	5770	5772	
5773	5776	5778	5783	5784	5815	5821	5822	5826	5827	5828	5831	5833	
5834	5837	5838	5841	5889	5891	5892	5895	5897	5902	5903	5934	5940	
5941	5945	5946	5947	5950	5952	5953	5956	5957	5960	6008	6010	6011	
6014	6016	6021	6022	6060	6067	6068	6072	6073	6074	6078	6081	6082	
6085	6086	6088	6138	6140	6141	6144	6146	6151	6152	6185	6192	6193	
6197	6198	6199	6202	6204	6205	6210	6211	6213	6295	6297	6298	6301	
6303	6309	6310	6332	6339	6340	6344	6345	6346	6349	6351	6352	6357	
6358	6360	6442	6444	6445	6448	6450	6456	6457	6483	6490	6491	6495	
6496	6497	6501	6504	6505	6508	6509	6511	6561	6563	6564	6567	6569	
6574	6575	6601	6605	6606	6678	6682	6683	6755	6761	6762	6873	6903	
6917	6932	6958	6970	6982	7007	7010	7011	7012	7013	7016	7017	7018	
7019													
1101#	6621	6698	6777										

DRA 000001



EM212	056527	1683	1689	8807#													
EM213	056577	1695	1701	1707	1713	1719	8814#										
EM214	056647	1725	1731	1737	1743	1773	882#										
EM215	056717	1749	1755	1761	1767	8828#											
EM216	057003	1779	1785	8837#													
EM217	057053	1791	1797	1803	1809	1839	1845	1851	8844#								
EM218	057106	1815	1821	1827	1833	8849#											
EM219	057166	1857	1863	1869	1875	1909	1915	1921	8857#								
EM220	057246	1882	1889	1896	1903	8865#											
EM221	057325	1932	8873#														
EM222	057402	1938	1944	8881#													
EM223	057455	1950	1956	1962	8889#												
EM224	057533	1968	8897#														
EM225	057630	1974	1980	8908#													
EM226	057715	1986	1992	1998	8917#												
EM227	060015	2004	2010	2016	2022	8928#											
EM230	060123	5175	8941#														
EM231	060231	2032	8953#														
EM232	060353	2037	8968#														
EM233	060475	5244	5364	5483	5602	5721	5840	5959	6087	6212	6359	8983#					
EM234	060541	5262	5382	5501	5620	5739	5858	5977	6105	6231	6378	8990#					
EM235	060601	2042	8996#														
EM236	060711	2047	9009#														
EM237	060773	6251	6398	9018#													
EM238	061041	6264	6411	9025#													
EM239	061100	6510	9031#														
EM240	061172	6528	9041#														
EM241	061261	2053	2059	2065	2071	9051#											
EM242	061345	2077	2083	2089	2095	9060#											
EM243	061431	2101	2107	2113	2119	9069#											
EM244	061532	2125	2131	2137	2143	9080#											
EM3000	061564	1312	1330	1348	1366	1480	1522	1546	1589	1612	1654	1684	1696	1726			
		1750	1780	1792	1816	1858	1883	1933	1939	2005	2048	2054	2078	2102			
		2126	9085#														
EM3001	061627	1318	1336	1354	1372	9091#											
EM3002	061653	1324	1342	1360	1378	9095#											
EM3003	061677	1384	1414	9099#													
EM3004	061747	1390	9106#														
EM3005	062021	1396	9114#														
EM3006	062123	1402	9126#														
EM3007	062224	1408	9138#														
EM3008	062345	1420	9153#														
EM3009	062425	1426	9162#														
EM3010	062504	1438	9170#														
EM3011	062551	1444	9177#														
EM3012	062620	1450	9184#														
EM3013	062666	1456	9191#														
EM3014	062723	1432	9196#														
EM3015	062774	1462	9203#														
EM3016	063042	1468	9210#														
EM3018	063107	1486	1528	1552	1594	1618	1660	1690	1702	1732	1756	1768	1786	1798			
		1822	1864	1890	2011	2060	2084	2108	2132	9217#							
EM3019	063153	1492	1534	1558	1600	1624	1672	1708	1738	1762	1904	1828	1870	1897			
		9224#															
EM3020	063201	1498	1540	1564	1606	1630	1678	1714	1744	1810	1834	1876	1904	9228#			
EM3021	063232	1504	1570	1636	1720	1840	1910	1963	1999	9233#							

EM3022	063256	1510	1576	1642	1846	1916	1951	1987	9237#					
EM3023	063326	1516	1582	1648	1852	1922	1957	1993	9244#					
EM3024	063376	1666	1774	9251#										
EM3025	063422	1474	9255#											
EM3026	063472	1939	1975	9262#										
EM3027	063541	1945	1981	9269#										
EM3028	063610	2017	9276#											
EM3029	063657	2023	9283#											
EM3030	063726	2033	2038	2043	9290#									
EM3031	063754	2066	2090	2114	2138	9294#								
EM3032	064007	2072	2096	2120	2144	9299#								
ERRCNT	003610	2212#	2286*	7411*	7414									
ERRVEC=	000004	956#	2267	2268*	2279*	7055	7056	7060*	7065*	7077*	7082*	7091*	7101*	7102*
		7133	7134*	7136*	7139*									
E.ASOF	003516	2175#												
E.BA	003504	2170#	2828*	2835	2868	2893	2925	2956	2987*	3035	3087*	3114	3120*	3128
		3151	3202*	3229	3241	3247*	3255*	3271	3331*	3358	3370	3376*	3385*	3404
		3434*	3447	3458*	3505*	3545	3573	3623*	3666	3754*	3762	3787*	3825	3831*
		3917*	3925	3950*	3988	3994*	4080*	4088	4113*	4151	4157*	4193*	4219	4234
		4240*	4249*	4261	8129	8134	8138	8154						
E.CS1	003500	2168#	2401*	2404	2446*	2449	2489*	2492	2533*	2536	2577*	2578	2635*	2636
		2666*	2667	2716*	2717	2747*	2748	2790*	2791	2827*	2830	2863	2888	2920
		2953	2988*	3029	3049	3079*	3122	3145	3170	3199*	3235	3265	3296	3328*
		3364	3398	3421	3441	3477	3502*	3539	3567	3597	3627*	3657	3675*	3677
		3727*	3756	3789*	3819	3890*	3919	3952*	3982	4053*	4082	4115*	4145	4194*
		4228	4255	4284	4339*	4351	4373*	4375	4390	4449*	4461	4483*	4485	4500
		4559*	4571	4593*	4595	4610	4669*	4681	4703*	4705	4720	4779*	4791	4813*
		4815	4830	4888*	4900	4922*	4924	4939	4983*	4989	5004*	5006	5057*	5069
		5091*	5093	5108	5311*	5312	5431*	5432	5550*	5551	5669*	5670	5788*	5789
		5907*	5908	6026*	6027	6156*	6157	6314*	6315	6461*	6462	6579*	6580	6619*
		6623	6645*	6649	6696*	6700	6722*	6726	6775*	6779	6801*	6805	8121	8124
		8129	8134	8138	8144	8148	8151	8156	8158	8160	8169	8173		
E.CS2	003510	2172#	3028*	3032	3048*	3052	3105*	3125	3148	3169*	3173	3220*	3238	3251*
		3268	3280*	3295*	3299	3349*	3367	3381*	3401	3410*	3424	3436*	3444	3460*
		3476*	3480	3524*	3542	3554*	3570	3583*	3596*	3600	3653*	3656*	3660	3676*
		3680	3753*	3759	3818*	3822	3916*	3922	3981*	3985	4079*	4085	4144*	4148
		4209	4227*	4231	4258	4268*	4283*	4287	4374*	4378	4389*	4395	4484*	4488
		4499*	4505	4594*	4598	4609*	4615	4704*	4708	4719*	4725	4814*	4818	4829*
		4835	4923*	4927	4938*	4944	4984*	4994	5005*	5009	5092*	5096	5107*	5113
		6620*	6626	6646*	6652	6697*	6703	6723*	6729	6776*	6782	6802*	6808	8138
		8144	8148	8151	8156	8160	8169	8173						
E.DA	003506	2171#												
E.DB	003522	2177#	3042*	3043	3161*	3162	3282*	3283	3412*	3413	3463*	3464	3580*	3584
		3769*	3770	3838*	3839	3932*	3933	4001*	4002	4095*	4096	4164*	4165	4272*
		4273	4383*	4400	4493*	4510	4603*	4620	4713*	4730	4823*	4840	4932*	4949
		5101*	5118	8142	8146									
E.DCYL	003520	2176#												
E.DS	003512	2173#	6621*	6629	6647*	6655	6698*	6706	6724*	6732	6777*	6785	6803*	6811
		8169	8173											
E.ECPS	003532	2181#												
E.ECPT	003534	2182#												
E.ER	003514	2174#	3652*	3663	6622*	6632	6648*	6658	6699*	6709	6725*	6735	6778*	6788
		6804*	6814	8151	8169	8173								
E.MR1	003524	2178#	2586*	2593	2602*	2606	2618	2628	2681*	2687	2699	2710	2755*	2761
		2773	2784	2845*	2853	2910	2952*	2962	5162*	5182*	5183*	5186	5189*	5190*
		5193	5245*	5301*	5302	5365*	5421*	5422	5484*	5540*	5541	5603*	5659*	5660

		5722*	5778*	5779	5841*	5897*	5898	5960*	6016*	6017	6088*	6146*	6147	6213*
		6277*	6283*	6303*	6304	6360*	6424*	6430*	6450*	6451	6511*	6569*	6570	6872*
		6875	6886*	6887*	6892*	6897*	6902*	6905	6916*	6919	6930*	6931*	6934	6951*
		6954*	6957*	6960	6969*	6972	6981*	6984	8126	8132	8134	8163	8167	
E.MR2	003526	2179#	2402*	2409	2447*	2454	2490*	2497	2534*	2541	8121			
E.MR3	003530	2180#	2403*	2412	2448*	2457	2491*	2500	2535*	2544	8121			
E.SPAR	003536	2183#												
E.WC	003502	2169#	2829*	2840	2873	2898	2930	2959	2989*	3038	3088*	3089*	3090	3121*
		3131	3134	3154	3204*	3244	3248*	3254*	3274	3333*	3373	3377*	3378	3384*
		3407	3435*	3450	3459*	3507*	3533	3548	3551*	3557*	3576	3625*	3669	3755*
		3765	3785*	3812	3828	3832*	3918*	3928	3948*	3975	3991	3995*	4081*	4091
		4111*	4136	4154	4158*	4191*	4225	4237	4241*	4250*	4264	8129	8134	8141
		8154												
FMTE	= 000020	1086#	6622	6699										
GNS	= ***** U	1148	2306	6842	6849	8103	8104	8105	8106	8107	8109	8111	8112	8113
		8114	8115	8116	8117									
GO	= 000001	1049#	3627	4373	4483	4593	4703	4813	4922	5004	5091	5311	5431	5550
		5669	5788	5907	6026	6156	6314	6461	6579	6619	6696	6775		
GTSWR	= 104406	2301	8109#											
HEAD1	064422	4338	4382	4887	4931	5056	5100	5220	5263	6061	6062	6484	6485	9353#
HEAD2	064420	4448	4492	5340	5383	9356#								
HEAD3	064426	4558	4602	5459	5502	6186	6187	9359#						
HEAD4	064442	4668	4712	5578	5621	9365#								
HEAD5	064450	4778	4822	5697	5740	9368#								
HEAD6	064456	5816	5859	9371#										
HEAD7	064464	5935	5978	9374#										
HT	= 000011	866#	7479	7537										
HVRC	= 000400	1090#												
IDAE	= 002000	1092#												
IE	= 000100	1050#												
ILF	= 000001	1082#												
INTR	= 000300	1045#												
IO'VEC	= 000020	961#	2251*	2252*										
IR	= 000100	1069#	3028	3048	3105	3169	3220	3280	3295	3349	3410	3460	3476	3524
		3583	3596	3653	3676	3753	3818	3916	3981	4079	4144	4209	4268	4283
		4374	4389	4484	4499	4594	4609	4704	4719	4814	4829	4923	4938	4984
		5005	5092	5107	6620	6646	6697	6723	6776	6802				
		994#	7041	7069										
KIPAR0	= 172340													
KIPAR1	= 172342	995#												
KIPAR2	= 172344	996#												
KIPAR3	= 172346	997#												
KIPAR4	= 172350	998#												
KIPAR5	= 172352	999#												
KIPAR6	= 172354	1000#	3716*	3774*	3879*	3937*	4042*	4100*						
KIPAR7	= 172356	1001#	7085											
KIPDR0	= 172300	983#												
KIPDR1	= 172302	984#												
KIPDR2	= 172304	985#												
KIPDR3	= 172306	986#												
KIPDR4	= 172310	987#												
KIPDR5	= 172312	988#												
KIPDR6	= 172314	989#												
KIPDR7	= 172316	990#												
LF	= 000012	867#	7531	7537										
MC_LK	= 000400	1123#	2394	2439	2482	2526	2572	2661	2675	2742	2820	2848	2850	2880
		2905	2907	2937	2944	3005	3011	3017	3094	3100	3107	3137	3209	3215





CZR6CE0 RK611 DSKLS CTRL PRT3  
 CZR6CE.P11 27-AUG-81 10:24

MACV11 30(1046) 28-AUG-81 10:35 PAGE 190  
 CROSS REFERENCE TABLE -- USER SYMBOLS

SEG C188

P.WC	003542	2188#												
P1.BIT	003612	2213#	5170*	5247*	5258*	5269	5272*	5275*	5286	5287*	5367*	5378*	5389	5392*
		5395*	5406	5407*	5486*	5497*	5508	5511*	5514*	5525	5526*	5605*	5616*	5627
		5630*	5633*	5644	5645*	5724*	5735*	5746	5749*	5752*	5763	5764*	5843*	5854*
		5865	5868*	5871*	5882	5883*	5962*	5973*	5984	5987*	5990*	6001	6002*	6090*
		6101*	6111	6114*	6117*	6128	6129*	6215*	6227*	6237	6240*	6243*	6255	6256*
		6261*	6269	6270*	6362*	6374*	6384	6387*	6390*	6402	6403*	6408*	6416	6417*
		6513*	6524*	6534	6537*	6540*	6551	6552*	6893	6947	8163			
RDBIT	043540	2591	2604	2616	2626	2685	2697	2708	2759	2771	2782	4332	4336	4349
		4368	4442	4446	4459	4478	4552	4556	4569	4588	4662	4666	4679	4698
		4772	4776	4789	4808	4881	4885	4898	4917	4986	5050	5054	5067	5086
		7003#												
RDCHR =	104410	7887	8112#											
RDDATA=	000021	1040#												
RDGATE=	100000	1130#	2602											
RDHEAD=	000025	1042#	2391	2401	2479	2489	2570	2577	2635	2659	2666	2716	2740	2747
		2790	4321	4339	4373	4431	4449	4483	4541	4559	4593	4651	4669	4703
		4761	4779	4813	4870	4888	4922	4973	4983	5004	5037	5057	5091	6603
		6619	6680	6696										
RDLIN =	104411	7961	8113#											
RDOCT =	104412	2317	2328	2345	8114#									
RDV =	000200	1051#	3627	3675	4373	4483	4593	4703	4813	4922	5004	5091	5311	5431
		5550	5669	5788	5907	6026	6156	6314	6461	6579	6619	6645	6696	6722
		6775	6801											
RECAL =	000013	1037#												
RESREG=	104414	7410	8116#											
RESTR	003652	1153	2231#											
RESVEC=	000010	957#												
RKASOF=	000016	1020#												
RKBA =	000004	1015#	2816*	2825	2861	2886	2918	2950	2999*	3022	3086*	3112	3143	3203*
		3227	3263	3332*	3356	3396	3439	3506*	3531	3565	3624*	3649	3747	3788*
		3810	3910	3951*	3973	4073	4114*	4136	4192*	4217	4253	5146*	5220*	5340*
		5459*	5578*	5697*	5816*	5935*	6061*	6186*	6333*	6484*	6758*			
RKCS1 =	000000	1013#	2386*	2391*	2398	2407*	2431*	2436*	2443	2452*	2474*	2479*	2486	2495*
		2518*	2523*	2530	2539*	2566*	2570*	2576	2581*	2596*	2609*	2621*	2631*	2634
		2655*	2656*	2659*	2665	2670*	2690*	2702*	2712*	2715	2736*	2737*	2740*	2746
		2751*	2764*	2776*	2786*	2789	2812*	2818*	2824	2860	2885	2917	2949	2965*
		2994*	3002*	3021	3046	3084*	3091*	3111	3141	3165	3200*	3206*	3226	3261
		3286	3329*	3335*	3355	3394	3416	3437	3467	3503*	3510*	3530	3563	3587
		3621*	3628*	3647	3672*	3673	3724*	3728*	3746	3783*	3790*	3809	3845*	3887*
		3891*	3909	3946*	3953*	3972	4008*	4050*	4054*	4072	4109*	4116*	4135	417*
		4188*	4195*	4216	4251	4279	4319*	4321*	4350	4358*	4371	4385	4429*	4431*
		4460	4468*	4481	4495	4539*	4541*	4570	4578*	4591	4605	4649*	4651*	4680
		4688*	4701	4715	4759*	4761*	4790	4798*	481*	4825	4868*	4870*	4899	4907*
		4920	4934	4971*	4973*	4987	5002	5012*	5035*	5037*	5068	5076*	5089	5103
		5142*	5148*	5218*	5222*	5310	5338*	5342*	5430	5457*	5461*	5549	5576*	5580*
		5668	5695*	5699*	5787	5814*	5818*	5906	5933*	5937*	6025	6059*	6064*	6155
		6184*	6189*	6313	6331*	6336*	6460	6482*	6487*	6578	6603*	6611	6615	6640*
		6641	6680*	6688	6692	6717*	6718	6759*	6767	6771	6796*	6797	7428*	
RKCS2 =	000010	1017#	2390*	2435*	2478*	2522*	3000*	3027	3047	3119	3142	3166	3234	3262
		3292	3363	3395	3420	3438	3473	3509*	3538	3564	3593	3648	3674	3752
		3817	3915	3980	4078	4143	4224	4252	4280	4372	4386	4482	4496	4592
		4606	4702	4716	4812	4826	4921	4935	4988	5003	5090	5104	6600*	6616
		6642	6677*	6693	6719	6754*	6772	6798						
RKDA =	000006	1016#	2389*	2434*	2477*	2521*	2998*							
RKDB =	000024	1022#	3041	3160	3281	3411	3462	3582	3768	3837	3931	4000	4094	4163







S.RECL=	000040	1135#																		
S.RTC =	000200	1137#	2447																	
S.SEEK=	000020	1134#	2402	2447																
S.STSP=	000100	1136#																		
S.UNLD=	002000	1140#																		
TBITVE=	000014	958#																		
TKVEC =	000060	965#	7703*	7704*																
TPVEC =	000064	966#																		
TRAPPC	003604	2207#	7025*	8120																
TRAPVE=	000034	964#	2255*	2256*																
TRTVEC=	000014	959#																		
TSTBY1	052443	3709	3872	4035	8421#															
TSTBY2	052453	3712	3875	4038	8423#															
TST1	004650	2367	2383#	7187																
TST10	007432	2752	2765	2777	2787	2792	2809#	7194												
TST11	010556	2982#	7195																	
TST12	011244	3058	3074#	7196																
TST13	012130	3182	3196#	7197																
TST14	013016	3325#	7198																	
TST15	014222	3499#	7199																	
TST16	015100	3618#	7200																	
TST17	015536	3681	3699#	7201																
TST2	005066	2408	2413	2428#	7188															
TST20	016652	3713	3862#	7202																
TST21	017766	3876	4025#	7203																
TST22	021102	4039	4185#	7204																
TST23	021772	4316#	7205																	
TST24	022524	4359	4393	4398	4426#	7206														
TST25	023256	4469	4503	4508	4536#	7207														
TST26	024010	4579	4613	4618	4646#	7208														
TST27	024542	4689	4723	4728	4756#	7209														
TST3	005304	2453	2458	2471#	7189															
TST30	025274	4799	4833	4838	4865#	7210														
TST31	026026	4908	4942	4947	4968#	7211														
TST32	026326	4992	4997	5032#	7212															
TST33	027074	5077	5111	5116	5139#	7213														
TST34	027464	5194	5215#	7214																
TST35	030310	5313	5335#	7215																
TST36	031134	5433	5454#	7216																
TST37	031760	5552	5573#	7217																
TST4	005520	2496	2501	2515#	7190															
TST40	032504	5671	5692#	7218																
TST41	033430	5790	5811#	7219																
TST42	034254	5909	5930#	7220																
TST43	035100	6028	6056#	7221																
TST44	035740	6158	6181#	7222																
TST45	036764	6316	6328#	7223																
TST46	040010	6463	6479#	7224																
TST47	040650	6581	6597#	7225																
TST5	005734	2540	2545	2563#	7191															
TST50	041274	6674#	7226																	
TST51	041720	6751#	7227																	
TST6	006416	2582	2597	2610	2622	2632	2637	2652#	7192											
TST7	007044	2671	2691	2703	2713	2718	2733#	7193												
TYPDS =	104405	6846	6853	8107#																
TYPE =	104401	2293	2310	2313	2316	2323	2327	2335	2344	3709	3712	3872	3875	4035						

CZR6CEO RK611 DSKLS CTRL PR13  
CZR6CE.P11 27-AUG-81 10:24

MACY11 30(1046) 28-AUG-81 10:35 PAGE 194  
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0192

		4038	6840	6847	6854	7315	7323	7369	7370	7374	7375	7382	7393	7395
		7405	7406	7408	7416	7430	7484	7597	7672	7725	7737	7791	7792	7795
		7806	7816	7827	7846	7894	7900	7905	7909	7914	7915	7917	7920	7924
		7990	7992	8064	8103#									
		7322	7359#											
		2315	2326	2343	3711	3874	4037	7390	7794	8104#				
TYPERR	045346	8106#												
TYPOC =	104402	8105#												
TYPON =	104404	2156#												
TYPOS =	104403	2151#	2825*	2835	2861*	2868	2886*	2893	2918*	2925	2950*	2956	3022*	3035
T.ASOF	003456	3112*	3128	3143*	3151	3227*	3241	3263*	3271	3356*	3370	3396*	3404	3439*
T.BA	003444	3447	3531*	3545	3565*	3573	3649*	3666	3747*	3762	3810*	3825	3910*	3925
		3973*	3988	4073*	4088	4136*	4151	4217*	4234	4253*	4261	8129	8134	8138
		8154												
T.CS*	003440	2149#	2398*	2404	2443*	2449	2486*	2492	2530*	2536	2576*	2578	2634*	2636
		2665*	2667	2715*	2717	2746*	2748	2789*	2791	2824*	2830	2860*	2863	2885*
		2888	2917*	2920	2949*	2953	3021*	3029	3046*	3049	3111*	3122	3141*	3145
		3165*	3170	3226*	3235	3261*	3265	3286*	3296	3355*	3364	3394*	3398	3416*
		3421	3437*	3441	3467*	3477	3530*	3539	3563*	3567	3587*	3597	3647*	3657
		3673*	3677	3746*	3756	3809*	3819	3909*	3919	3972*	3982	4072*	4082	4135*
		4145	4216*	4228	4251*	4255	4279*	4284	4350*	4351	4371*	4375	4385*	4390
		4460*	4461	4481*	4485	4495*	4500	4570*	4571	4591*	4595	4605*	4610	4680*
		4681	4701*	4705	4715*	4720	4790*	4791	4811*	4815	4825*	4830	4899*	4900
		4920*	4924	4934*	4939	4987*	4989	5002*	5006	5068*	5069	5089*	5093	5103*
		5108	5310*	5312	5430*	5432	5549*	5551	5668*	5670	5787*	5789	5906*	5908
		6025*	6027	6155*	6157	6313*	6315	6460*	6462	6578*	6580	6615*	6623	6635
		6641*	6649	6692*	6700	6712	6718*	6726	6771*	6779	6791	6797*	6805	8121
		8124	8129	8134	8138	8144	8148	8151	8156	8158	8160	8169	8173	
T.CS2	003450	2153#	3027*	3032	3047*	3052	3119*	3125	3142*	3148	3166*	3173	3234*	3238
		3262*	3268	3292*	3299	3363*	3367	3395*	3401	3420*	3424	3438*	3444	3473*
		3480	3538*	3542	3564*	3570	3593*	3600	3648*	3654	3660	3674*	3680	3752*
		3759	3817*	3822	3915*	3922	3980*	3985	4078*	4085	4143*	4148	4224*	4231
		4252*	4258	4280*	4287	4372*	4378	4386*	4395	4482*	4488	4496*	4505	4592*
		4598	4606*	4615	4702*	4708	4716*	4725	4812*	4818	4826*	4835	4921*	4927
		4935*	4944	4988*	4994	5003*	5009	5090*	5096	5104*	5113	6616*	6626	6636
		6642*	6652	6693*	6703	6713	6719*	6729	6772*	6782	6792	6798*	6808	8138
		8144	8148	8151	8156	8169	8173							
T.DA	003446	2152#												
T.DB	003462	2158#	3041*	3043	3160*	3162	3281*	3283	3411*	3413	3462*	3464	3582*	3584
		3768*	3770	3837*	3839	3931*	3933	4000*	4002	4094*	4096	4163*	4165	4271*
		4273	4384*	4400	4494*	4510	4604*	4620	4714*	4730	4824*	4840	4933*	4949
		5102*	5118	8142	8146									
T.DCYL	003460	2157#												
T.DS	003452	2154#	6617*	6629	6637	6643*	6655	6694*	6706	6714	6720*	6732	6773*	6785
		6793	6799*	6811	8169	8173								
T.ECPS	003472	2162#												
T.ECPT	003474	2163#												
T.ER	003454	2155#	3651*	3663	6618*	6632	6638	6644*	6658	6695*	6709	6715	6721*	6735
		6774*	6788	6794	6800*	6814	8151	8169	8173					
T.MR1	003464	2159#	2592*	2593	2605*	2606	2617*	2618	2627*	2628	2686*	2687	2698*	2699
		2709*	2710	2760*	2761	2772*	2773	2783*	2784	2852*	2853	2909*	2910	2948*
		2962	5185*	5186	5192*	5193	5300*	5302	5420*	5422	5539*	5541	5658*	5660
		5777*	5779	5896*	5898	6015*	6017	6145*	6147	6302*	6304	6449*	6451	6568*
		6570	6874*	6875	6904*	6905	6918*	6919	6933*	6934	6959*	6960	6971*	6972
		6983*	6984	8126	8132	8134	8163	8167						
T.MR2	003466	2160#	2399*	2409	2444*	2454	2487*	2497	2531*	2541	8121			

CZR6CEO RK611 DSKLS CTRL PRT3  
CZR6CE.P11 27-AUG-81 10:24

MACY11 30(1046) 28-AUG-81 10:35 PAGE 195  
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0193

T.MR3	003470	2161#	2400*	2412	2445*	2457	2488*	2500	2532*	2544	8121			
T.SPAR	003476	2161#												
T.WC	003442	2161#	2826*	2840	2862*	2873	2887*	2898	2919*	2930	2951*	2959	3023*	3038
		3113*	3131	3144*	3154	3228*	3244	3264*	3274	3357*	3373	3397*	3407	3440*
		3450	3532*	3548	3566*	3576	3650*	3669	3748*	3765	3811*	3828	3911*	3928
		3974*	3991	4074*	4091	4137*	4154	4218*	4237	4254*	4264	8129	8134	8141
		8154												
JFE =	000400	1071#												
UNLOAD=	000007	1035#												
UNS =	040000	1096#												
UPE =	020000	1076#												
VV =	000100	1107#												
WAITIM	003632	2221#	6610	6687	6766									
WCE =	040000	1077#												
WLE =	004000	1093#												
WRBUFF	064714	2816	2828	2987	2995*	2999	5146	6758	9450#	9451				
WRDATA=	000023	1041#												
WRDCNT	003624	2218#	3159*	3176*	3279*	3287	3302*	3387*	3453*	3468	3483*	3581*	3588	3603*
		4270*	4290*	4353*	4354*	4381*	4387	4403*	4404	4463*	4464*	4491*	4497	4513*
		4514	4573*	4574*	4601*	4607	4623*	4624	4683*	4684*	4711*	4717	4733*	4734
		4793*	4794*	4821*	4827	4843*	4844	4902*	4903*	4930*	4936	4952*	4953	5071*
		5072*	5099*	5105	5121*	5122	8146	8148	8158					
WRHEAD-	000027	1043#	2436	2446	2523	2533	2818	2827	2988	3002	3079	3091	3199	3206
		3328	3335	3502	3510	3627	3628	3727	3728	3789	3790	3890	3891	3952
		3953	4053	4054	4115	4116	4194	4195	5148	5222	5311	5342	5431	5461
		5550	5580	5669	5699	5788	5818	5907	5937	6026	6064	6156	6189	6314
		6336	6461	6487	6579	6759	6775							
WRL =	004000	1110#												
WRFBIT	042570	5177	5253	5259	5276	5288	5373	5379	5396	5408	5492	5498	5515	5527
		5611	5617	5634	5646	5730	5736	5753	5765	5849	5855	5872	5884	5968
		5974	5991	6003	6096	6102	6118	6130	6222	6228	6244	6257	6262	6271
		6278	6285	6369	6375	6391	6404	6409	6418	6425	6432	6519	6525	6541
		6553	6872#											
WRCHK=	000031	1044#												
WRGAT=	040000	1121#	2952	5162	5182	5190	5245	5365	5484	5603	5722	5841	5960	6088
		6213	6360	6511	6887									
WR.PAR=	000004	1005#												
\$APTND	001000	1179	1185#											
\$ASTAT-	***** U	7271	7286											
\$ATYC	044724	7242	7244#											
\$ATY1	044700	7240#												
\$ATY3	044706	7241#	7469											
\$ATY4	044716	7243#	7328											
\$AUTOB	001134	1216#	2303*	7784	7941									
\$BASE	001270	1288#	2314	2322*	2363	2385	2430	2473	2517	2565	2654	2735	2811	2984
		3076	3198	3327	3501	3620	3715	3878	4041	4187	4318	4428	4538	4648
		4758	4867	4970	5034	5141	5217	5337	5456	5575	5694	5813	5932	6058
		6183	6330	6481	6599	6676	6753	7427	8066					
\$BDADR	001122	1211#												
\$BDDAT	001126	1213#												
\$BELL	001204	1239#	7315	7348	7737	7934								
\$CDW1	001274	1290#												
\$CDW2	001276	1291#												
\$CHARC	046236	7486*	7496*	7503	7529*	7534#								
\$CKSWR	047224	7770#	8111											
\$CMTAG	001100	1199#	2244	2245	2253	2259	2260	2261						















CZR6CEO RK611 DSK!S CTRL PR13  
CZR6CE.P11 27-AUG-81 10:24

MACV11 30(1046) 28-AUG-81 10:35 PAGE 203  
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0200

.\$MULT	1#		
.\$POWE	1#	835#	
.\$RAND	1#		
.\$RDDE	1#		
.\$RDOC	1#	835#	7942
.\$READ	1#	835#	7681
.\$R2AZ	1#		
.\$SAVE	1#	835#	7995
.\$SB2D	1#		
.\$SB2O	1#		
.\$SCOP	1#	835#	7112
.\$SIZE	1#	835#	7035
.\$SUPR	1#		
.\$TRAP	1#	835#	8072
.\$TYPB	1#		
.\$TYPD	1#	835#	7614
.\$TYPE	1#	835#	7441
.\$TYPO	1#	835#	7537
.\$4OCA	1#		
.1170	1#		

. ABS. 065120 000

ERRORS DETECTED: 0

CZR6CE,CZR6CE.LST/SOL/CRF/NL:TOC=CZR6CE.SML,CZR6CE.P11  
RUN-TIME: 32 39 3 SECONDS  
RUN-TIME RATIO: 254/75=3.3  
CORE USED: 44K (88 PAGES)