

RP04/05/06

CONTROLLER LOGIC PART 2  
MD-11-DZRJF-A

EP-DZRJF-A-DL-A

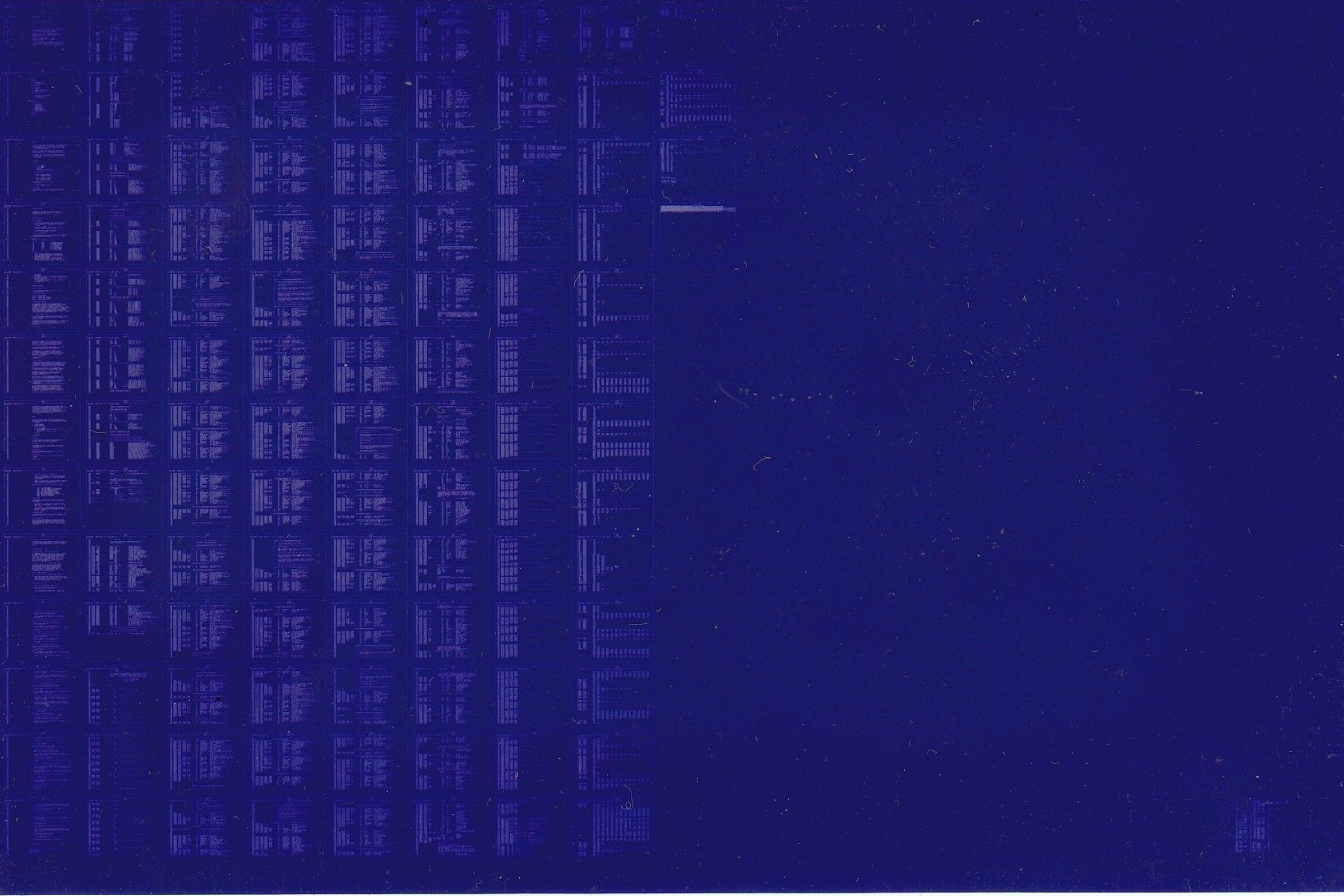
NOV 1978

COPYRIGHT © 1978



FICHE 1 OF 1

MADE IN U.S.A.





801

MC-11-DZRJF-A, RPO4/5/6 DUAL CONTRLR LOGIC TEST - PART 2  
DZRJFA.CMB 02-NOV-76 19:11

MACY11 27(1006) 02-NOV-76 19:12 PAGE 1

.REM 2

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZRJF-A-D  
PRODUCT NAME: RPO4/5/6 DUAL CONTROLLER LOGIC TEST - PART 2  
DATE CREATED: MARCH 29, 1976  
MAINTAINER: DIAGNOSTIC ENGINEERING  
AUTHOR: U. NESS

COPYRIGHT (C) 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

THE INFORMATION IN THIS STATEMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

ACTUAL DISTRIBUTION OF THE SOFTWARE DESCRIBED IN THIS DOCUMENT WILL BE SUBJECT TO TERMS AND CONDITIONS TO BE ANNOUNCED ON SOME FUTUF. DATE BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE TO USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DEC.

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40

CONTENTS  
-----

- 1. ABSTRACT
- 2. REQUIREMENTS
  - 2.1 EQUIPMENT
  - 2.2 PRELIMINARY PROGRAMS
  - 2.3 OTHER PROGRAMS
- 3. LOADING PROCEDURES
- 4. STARTING PROCEDURES
  - 4.1 STARTING ADDRESSES
  - 4.2 UNIBUS & VECTOR ADDRESSES
  - 4.3 OPERATOR ACTION
- 5. OPERATING PROCEDURES
  - 5.1 OPERATIONAL SWITCH SETTINGS
  - 5.2 'SOFTWARE' SWITCH REGISTER
  - 5.3 TEST SELECTION
  - 5.4 DUAL PORT TEST CABLE CONNECTION
- 6. ERRORS
- 7. MISCELLANEOUS
  - 7.1 RESTRICTIONS
  - 7.2 LIMITATIONS
  - 7.3 EXECUTION TIME
  - 7.4 STACK POINTER
  - 7.5 SUBROUTINE CALLS
  - 7.6 REQUIRED TESTS
  - 7.7 DISK SURFACE USAGE
  - 7.8 LOOP ON ERROR OPTION
  - 7.9 SPINDLE MOTOR 'COOL DOWN' STALL
- 8. TEST DESCRIPTIONS
- 9. PROGRAM LISTING

4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63

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

1. ABSTRACT

THE RPO4/5/6 DUAL CONTROLLER OPTION LOGIC TEST PERFORMS A SERIES OF TESTS WHICH VERIFY THAT THE RPO4/5/6 DUAL CONTROLLER LOGIC IS FUNCTIONING PROPERLY. ONLY THE CONTROL LOGIC IS TESTED BY THIS PROGRAM; DATA HANDLING IN THE DUAL CONTROLLER MODE IS NOT TESTED BY THIS PROGRAM.

BOTH PORTS OF THE DRIVE ARE CABLED TO THE SAME MASSBUS BY A SPECIAL ADAPTER CABLE. THIS ARRANGEMENT ALLOWS THE DUAL CONTROLLER LOGIC TO BE TESTED FROM ONE PDP-11/RH11 OR RH70.

THIS PROGRAM IS THE SECOND PART OF THE RPO4 DUAL CONTROLLER OPTION LOGIC TEST. ONLY THE CONTROL LOGIC ASSOCIATED WITH THE UNLOAD COMMAND AND 'CONTROLLER SELECT' SWITCH IS TEST BY THIS PROGRAM.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR  
16K OF MEMORY  
KH11-L OR KH11-P CLOCK  
TELETYPE  
RH11 OR RH70 WITH AN RPO4/5/6  
RPO4/5/6 DUAL CONTROLLER OPTION TEST CABLE

2.2 PRELIMINARY PROGRAMS

A. RPO4/5/6 DISKLESS CONTROLLER TEST  
PART 1 (MAINDEC-11-DZRJG)  
PART 2 (MAINDEC-11-DZRJH)

B. RPO4/5/6 FUNCTIONAL CONTROLLER TEST  
PART 1 (MAINDEC-11-DZRJI)  
PART 2 (MAINDEC-11-DZRJJ)

THE PRELIMINARY PROGRAMS MUST BE RUN TWICE: ONCE FROM EACH CONTROLLER (PORT).

C. RPO4 DUAL CONTROLLER LOGIC TEST  
PART 1 (MAINDEC-11-DZRJE)

2.3 OTHER PROGRAMS

DYNAMIC OPERATION OF THE DUAL CONTROLLER OPTION IS TESTED BY THE RPO4/5/6 MULTIDRIVE EXERCISER PROGRAM (MAINDEC-11-DZRJD).

140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
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

3. LOADING PROCEDURES

THE PROGRAM MAY BE LOADED BY THE ABSOLUTE PAPER TAPE LOADER OR IT MAY BE LOADED FROM THE APPROPRIATE MEDIA USING THE ASSOCIATED 'XXDP' LOADER. THE PROGRAM MAY NOT BE INCLUDED IN AN 'XXDP' CHAIN.

4. STARTING PROCEDURES

4.1 STARTING ADDRESSES

- A. THE NORMAL STARTING ADDRESS OF THE PROGRAM IS LOCATION 200(B). STARTING AT THIS ADDRESS ALLOWS THE OPERATOR TO SELECT (OR RESELECT) THE ADDRESS OF THE DRIVE TO BE TESTED.
- B. THE RESTART ADDRESS IS LOCATION 204(B). THE PROGRAM WILL USE THE CURRENT DRIVE (DCL) ADDRESS.
- C. THE PROGRAM CAN BE STARTED AT LOCATION 210(B) TO ALLOW THE RH11 OR RH70 ADDRESS TO BE CHANGED.

4.2 UNIBUS & VECTOR ADDRESS

THE PROGRAM ASSUMES THE FOLLOWING UNIBUS AND VECTOR ADDRESSES. THESE ADDRESSES MAY BE CHANGED PRIOR TO STARTING THE PROGRAM FROM ANY OF THE STARTING LOCATIONS.

<u>MEMORY LOCATION</u>	<u>CONTENTS</u>	<u>FUNCTION</u>
1142	177560	TTY KEYBOARD STATUS REG
1144	177562	TTY KEYBOARD BUFFER REG
1146	177564	TTY PRINTER STATUS REG
1150	177566	TTY PRINTER BUFFER REG
1210	172540	KW11-P STATUS REG
1212	172542	KW11-L COUNTER BUFFER
1214	104	KW11-P VECTOR ADDRESS
1216	177546	KW11-L STATUS REGISTER
1220	100	KW11-L VECTOR ADDRESS

4.3 OPERATOR ACTION

- A. CONNECT THE DUAL CONTROLLER TEST CABLE BETWEEN BUS A & BUS B ON THE DRIVE BEING TESTED. (SEE SECTION 5.4)
- B. LOAD THE PROGRAM INTO MEMORY IN THE PROCESSOR CONTROLLING THE MASSBUS USED FOR TESTING.
- C. SWITCH THE 'CONTROLLER SELECT' SWITCH ON THE DRIVE TO BE TESTED TO THE 'A/B' POSITION. CYCLE THE DRIVE UP.
- D. LOAD THE APPROPRIATE STARTING ADDRESS (200(B) OR 210(B)) INTO THE SWITCH REGISTER (OR THE 'SOFTWARE' SWITCH REGISTER,

196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251

- SEE SECTION 5.2.)
- E. PRESS START.
  - F. ENTER THE DRIVE NUMBER. (THIS MUST BE THE NUMBER DISPLAYED BY THE DRIVE, IF AN RPO4, OR THE NUMBER OF ADDRESS PLUG IF THE DRIVE IS AN RPO5/6.)
  - G. ENTER THE NUMBER OF THE TEST TO BE RUN. ('CARRIAGE RETURN' OR '0' WILL RUN ALL TESTS.)
  - H. THE PROGRAM MAY BE STOPPED AT ANY TIME AND RESTARTED FROM LOCATION 204.

## 5. OPERATING PROCEDURES

-----

### 5.1 OPERATIONAL SWITCH SETTINGS

WITH ALL SWITCHES SET TO ZERO, THE PROGRAM WILL TYPE ALL ERRORS AND CONTINUE TESTING.

THE SWITCH SETTINGS ARE:

SW(15)=1...HALT ON ERROR  
 SW(14)=1...LOOP ON TEST  
 SW(13)=1...INHIBIT ERROR TYPEOUTS  
 SW(11)=1...INHIBIT TEST ITERATIONS  
 SW(10)=1...RING TTY BELL ON ERROR  
 SW(09)=1...LOOP ON ERROR

### 5.2 'SOFTWARE' SWITCH REGISTER

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RPO4/5/6 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 SETTING ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED. 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

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

### 5.3 TEST SELECTION

252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
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

INDIVIDUAL TESTS ARE SELECTED IN RESPONSE TO THE 'ENTER TEST NUMBER:' MESSAGE. ANY VALID TEST NUMBER CAN BE ENTERED. EACH ENTRY MUST BE TERMINATED BY A CARRIAGE RETURN (CR). THE LOOP ON TEST SWITCH, SW<15>, MUST BE SET TO ALL CONTINUOUS EXECUTION OF THE SELECTED TEST.

TO RUN ALL TESTS IN SEQUENCE, ENTER EITHER A '0' FOLLOWED BY A CARRIAGE RETURN OR A CARRIAGE RETURN BY ITSELF. THE PROGRAM WILL THEN EXECUTE ALL TESTS IN SEQUENCE.

THE 'RUBOUT KEY' (RO) CAN BE USED TO DELETE THE LAST CHARACTER ENTERED. SUCCESSIVELY STRIKING THE RO KEY WILL DELETE CHARACTERS UNTIL THE PREVIOUS CHARACTERS HAVE BEEN DELETED. CHARACTERS DELETED BY THE RO KEY WILL BE TYPED AND WILL BE SEPARATED BY '\' FROM THE CHARACTERS ENTERED BY THE OPERATOR.

THE OPERATOR CAN DELETE AN ENTIRE ENTRY BY TYPING A 'CONTROL U' (↑U).

5.4 TEST CABLE CONNECTION

TO TEST THE RPO4/5/6 DUAL CONTROLLER OPTION WITH THIS PROGRAM, A SPECIAL TEST CABLE MUST BE USED. (THE TEST CABLE IS P/N 7010507-02). THE TEST CABLE CONNECTS MASSBUS A & MASSBUS B TOGETHER AT THE DRIVE BEING TESTED AND IS CONSTRUCTED SO THAT BIT 0 OF THE MASSBUS UNIT SELECT LINES IS COMPLEMENTED.

WITH THE TEST CABLE CONNECTED TO THE DRIVE UNDER TEST, THE DRIVE APPEARS AS TWO UNITS ON THE MASSBUS: EACH PORT OF THE RPO4 WILL RESPOND TO A DIFFERENT MASSBUS ADDRESS.

THE ADDRESS OF EACH PORT WILL DEPEND UPON THE DRIVE'S ADDRESS (THE ADDRESS SELECTED BY THE SWITCHES ON THE 'DP' BOARD - MODULE M7775 FOR RPO4'S, OR BY THE ADDRESS PLUG FOR RPO5/6'S).

THE PROGRAM WILL TYPEOUT THE APPARENT ADDRESSES OF BOTH PORTS (ONE PORT WILL HAVE THE ADDRESS OF THE DRIVE; THE OTHER PORT WILL HAVE THE ADDRESS DEVELOPED BY THE CABLE).

\*\*\*\*\*  
\* ANY OTHER DRIVE ON THE MASSBUS WHICH HAS AN ADDRESS IN \*  
\* CONFLICT WITH EITHER OF THE TEST ADDRESSES MUST BE \*  
\* POWERED DOWN. \*  
\*\*\*\*\*

THE TEST CABLE CONNECTION TO THE DRIVE UNDER TEST WILL DEPEND ON WHICH PROCESSOR/RH11 IS TO TEST THE DRIVE. IF THE DRIVE IS TO BE TESTED BY THE PROCESSOR ON PORT A, THE TEST CABLE IS CONNECTED FROM 'BUS A OUT' TO 'BUS B IN'. IF THE DRIVE IS TO BE TESTED BY THE PORT B PROCESSOR, THE TEST CABLE IS CONNECTED FROM 'BUS B OUT' TO 'BUS A IN'.

WHEN THE DUAL PORT TEST CABLE IS CONNECTED, THE ATTENTION

308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
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

BITS FOR PORTS A & B ARE ASSERTED IN THE SAME BIT POSITION WHEN 'RPAS' (ATTENTION SUMMARY REGISTER) IS READ. THE ATTENTION BIT POSITION IS DETERMINED BY THE ADDRESS OF THE DRIVE. THE ATTENTION BIT THAT APPEARS FOR THE DRIVE IS THE INCLUSIVE 'OR' OF THE PORT A & PORT B ATTENTION BITS. BECAUSE OF THIS, THE PROGRAM LOOKS AT ONLY THE ATTENTION BIT IN 'RPSI' (DRIVE STATUS REGISTER) TO DETERMINE THE STATE OF THE SELECTED PORT'S ATTENTION BIT.

6. ERRORS

-----

WHEN THE PROGRAM ENCOUNTERS AN ERROR, THE ERROR ROUTINE IS CALLED AND IF SW(13) IS NOT SET, THE ERROR MESSAGE PERTAINING TO THE ERROR WILL BE TYPED. EACH ERROR TIMEOUT WILL CONTAIN THE FOLLOWING:

- A. AN ERROR MESSAGE
- B. A DATA HEADER LINE
- C. A DATA LINE CONTAINING:
  - 1. THE TEST NUMBER
  - 2. THE PC (PROGRAM COUNTER VALUE) WHERE THE ERROR CALL WAS MADE
  - 3. CONTENTS OF THE APPROPRIATE REGISTERS

7. MISCELLANEOUS

-----

7.1 RESTRICTIONS

TO RUN THIS PROGRAM, THE SYSTEM MUST HAVE EITHER A KW11-P OR A KW11-L CLOCK. ADDITIONALLY, THE RPO4 UNDER TEST MUST HAVE THE DUAL PORT TEST CABLE CONNECTED.

7.2 LIMITATIONS

THIS PROGRAM DOES NOT TEST DATA TRANSFERS THROUGH EITHER PORT AND DOES NOT TEST THE DYNAMIC OPERATION OF THE DUAL CONTROLLER OPTION.

7.3 EXECUTION TIME

THE PROGRAM TAKES ABOUT 60 SECONDS PER PASS (DEPENDING ON OPERATOR INTERVENTION EFFICIENCY).

7.4 STACK POINTER

THE STACK IS INITIALLY SET TO 1100 AND EXTENDS DOWNWARD IN MEMORY.

7.5 SUBROUTINE CALLS



364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
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

THE SUBROUTINE CALLS USED BY THE PROGRAM ARE:

- A. 'SCOPE' (IOT INSTRUCTION). THIS CALL IS PLACED BETWEEN EACH TEST IN THE INSTRUCTION. THIS ROUTINE ESTABLISHES THE ITERATION COUNT AND THE LOOP ON TEST AND LOOP ON ERROR ADDRESSES.
- B. 'ERROR' (EMT INSTRUCTION). THIS CALL IS USED TO REPORT ALL ERRORS. THE CALL IS FOLLOWED BY A NUMBER WHICH IDENTIFIES THE ERROR MESSAGE WHICH WILL BE TYPED.

THE TRAP INSTRUCTION IS USED FOR THE FOLLOWING SUBROUTINE CALLS:

- TYPE - TTY TYPEOUT ROUTINE
- TYPOC - TYPE OCTAL NUMBER (WITH LEADING ZERO)
- TYPOS - TYPE OCTAL NUMBER (NO LEADING ZEROS)
- TYPON - TYPE OCTAL NUMBER PER LAST CALL
- TYPOS - TYPE DECIMAL NUMBER WITH SIGN
- ROCHR - READ CHARACTER FROM TTY KEYBOARD
- ROLIN - READ A LINE FROM THE TTY KEYBOARD.
- RDOCT - READ AN OCTAL NUMBER FROM THE TTY KEYBOARD
- SAVREG - ROUTINE TO SAVE RO-RS
- RESREG - ROUTINE TO RESTORE RO-RS

7.6 REQUIRED TESTS

IF THE PROGRAM IS BEING EXECUTED IN SINGLE TEST MODE, THE OPERATOR MUST CALL AND RUN THE FOLLOWING TESTS BEFORE OTHER TESTS ARE RUN:

- A. TEST 2 AND TEST 3. THESE TESTS SET 'VV-A' AND 'VV-B' RESPECTIVELY. THESE TESTS MUST BE PERFORMED AT LEAST ONCE BEFORE TESTS 4 - 14 ARE RUN.
- B. TEST 4 AND TEST 5. THESE TESTS DETERMINE AND STORE FOR LATER USE THE TIMEOUT ONE-SHOT VALUE MEASURED THROUGH EACH PORT. THESE TESTS MUST BE PERFORMED AT LEAST ONCE BEFORE TESTS 6 - 14 ARE RUN.

7.7 DISK SURFACE USAGE

THE DIAGNOSTIC DOES NOT USE THE DISK SURFACE. HOWEVER, THE DRIVE MUST BE CYCLED UP AND ON LINE FOR THE DIAGNOSTIC TO BE RUN.

7.8 LOOP ON ERROR OPTION

IF SW<09> IS SET, THE PROGRAM WILL LOOP ON A FAILING TEST UNTIL EITHER THE SWITCH IS RESET OR THE ERROR STOPS OCCURRING. BECAUSE THE PROGRAM MUST RESET THE RPO4 TO A KNOWN STATE BEFORE LOOPING ON THE ERROR, THE TEST FOR SW<09> IS PERFORMED AT THE END OF THE TEST - NOT AT THE POINT WHERE THE ERROR WAS DETECTED.

420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475

7.9 SPINDLE MOTOR 'COOL DOWN' STALL

THE PROGRAM STALLS FOR 5 SECONDS AFTER THE SPINDLE MOTOR ON THE DRIVE BEING TESTED HAS BEEN CYCLED UP. THIS STALL ALLOWS THE MOTOR TO COOL DOWN AND WAS INCLUDED TO MINIMIZE THE NUMBER OF TIME THAT THE THERMAL BREAKER ON THE MOTOR WILL TRIP DURING TESTING.

8. TEST DESCRIPTIONS  
 -----

8.1 METHOD USED TO VERIFY THAT DRIVE IS IN NEUTRAL

THE PROGRAM DETERMINES THE THE DRIVE IS IN NEUTRAL BY CHECKING THE CONTENTS OF THE DRIVE STATUS REGISTER (RPS1) THROUGH BOTH PORTS. THE PROGRAM MASKS OUT THE PORT DEPENDENT BITS ('ATA' & 'VV') AND VERIFIES THAT CORRECT STATUS IS READ THROUGH BOTH PORTS. ( THE CORRECT STATUS IS 'MOL', 'PGM', 'DPR', & 'DRY'.) IF NEITHER PORT SEES ALL ZEROS FROM RPS1, THE PROGRAM CONCLUDES THAT THE DRIVE IS IN NEUTRAL AND THAT ANY BIT DISCREPANCY BETWEEN PORTS INDICATES A FAILURE IN THE PATH FOR THAT BIT.

8.2 METHOD USED TO VERIFY THAT THE DRIVE HAS BEEN SEIZED

THE PROGRAM VERIFIES THAT THE DRIVE HAS BEEN SEIZED BY CHECKING THE DRIVE STATUS REGISTER (RPS1) THROUGH THE SEIZING PORT AND VERIFYING THAT CORRECT STATUS IS SEEN. WHEN RPS1 IS READ THROUGH THE OPPOSITE PORT, ZEROS SHOULD BE SEEN. IF BOTH CONDITIONS EXIST. (I.E. THE OPPOSITE PORT), THE PROGRAM CONCLUDES THAT THE DRIVE HAS BEEN SEIZED BY THE SPECIFIED PORT.

8.3 TEST 1 - DRIVE ACCESS TEST

VERIFY THAT THE DRIVE CAN BE ACCESSED THROUGH BOTH PORTS

- A. SELECT DRIVE, VERIFY THAT THE DRIVE IS PRESENT, THAT THE DRIVE IS A DUAL PORT RPO4, THAT THE DRIVE IS ONLINE (RPS1 HAS 'MOL', 'PGM', 'DPR', & 'DRY' BITS SET), AND THE DRIVE SERIAL NUMBER READ THROUGH BOTH PORTS IS THE SAME.
- B. THE TEST IS REPEATED THROUGH BOTH PORTS.

8.4 TEST 2 - SET 'VV' FOR PORT 'A'

SET VOLUME VALID

- A. ISSUE A DRIVE CLEAR COMMAND THROUGH PORT 'A'.
- B. ISSUE A READIN PRESET COMMAND THROUGH PORT 'A'. VERIFY THAT THE 'VV' BIT IS SET FOR PORT 'A'.
- C. ISSUE A RELEASE COMMAND THROUGH PORT 'A'. VERIFY THAT THE DRIVE RETURNED TO NEUTRAL AND THAT NEITHER ATTENTION

476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531

BIT IS SET.

- 8.5 TEST 3 - SET 'VV' FOR PORT 'B'  
SET VOLUME VALID
- A. ISSUE A DRIVE CLEAR COMMAND THROUGH PORT 'B'.
  - B. ISSUE A READIN PRESET COMMAND THROUGH PORT 'B'. VERIFY THAT THE 'VV' BIT IS SET FOR PORT 'B'.
  - C. ISSUE A RELEASE COMMAND THROUGH PORT 'B'. VERIFY THAT THE DRIVE RETURNED TO NEUTRAL AND THAT NEITHER ATTENTION BIT IS SET.
- 8.6 TEST 4 - MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT 'A'  
MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT 'A'
- A. WRITE 0'S INTO RPOS1 THROUGH PORT 'A' AND VERIFY THAT THE DRIVE HAS BEEN SEIZED.
  - B. WAIT FOR TIMEOUT TO OCCUR. MEASURE THE DURATION OF THE TIMEOUT ONE-SHOT AND SAVE THE VALUE FOR LATER USE.
  - C. VERIFY THAT THE TIMEOUT OCCURED AND THAT THE DRIVE RETURNS TO NEUTRAL.
- 8.7 TEST 5 - MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT 'B'  
MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT 'B'
- A. WRITE 0'S INTO RPOS1 THROUGH PORT 'B' AND VERIFY THAT THE DRIVE HAS BEEN SEIZED.
  - B. WAIT FOR TIMEOUT TO OCCUR. MEASURE THE DURATION OF THE TIMEOUT ONE-SHOT AND SAVE THE VALUE FOR LATER USE.
  - C. VERIFY THAT THE TIMEOUT OCCURED AND THAT THE DRIVE RETURNS TO NEUTRAL.
- 8.8 TEST 6 - TEST UNLOAD COMMAND THROUGH PORT 'A'  
VERIFY THAT THE UNLOAD COMMAND FUNCTIONS PROPERLY AND THAT A PORT TIMEOUT WILL NOT OCCUR WHILE THE 'GO' BIT IS SET.
- A. ISSUE AN UNLOAD COMMAND THROUGH PORT 'A'; VERIFY THAT THE DRIVE IS SEIZED.
  - B. WAIT THE MEASURED TIMEOUT INTERVAL + 25%; VERIFY THAT THE DRIVE DOES NOT TIME OUT. VERIFY THAT THE 'GO' BIT IS STILL SET AND THAT 'DRY' AND 'PIP' ARE NOT SET.
  - C. REQUEST THAT THE OPERATOR PRESS THE 'STANDBY' BUTTON ON THE DRIVE.
  - D. WHEN THE DRIVE CYCLES UP, VERIFY THAT THE DRIVE IS STILL SEIZED

532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587

BY PORT A, THAT THE 'VV' BIT FOR PORT A IS RESET, THAT THE ATTENTION BIT FOR PORT 'A' IS SET, AND THAT THE ATTENTION BIT FOR PORT 'B' IS NOT SET.

- E. WAIT FOR THE PORT TIMEOUT TO RELEASE THE DRIVE. WHEN THE TIMEOUT OCCURS, VERIFY THAT THE DRIVE RETURNED TO NEUTRAL, THAT THE ATTENTION BIT FOR PORT 'A' IS SET, AND THAT THE ATTENTION BIT BIT FOR PORT 'B' IS NOT SET.
- F. VERIFY THAT THE 'VV' BIT FOR PORT 'B' IS NOT SET.
- G. ISSUE A PACK ACKNOWLEDGE INSTRUCTION THROUGH BOTH PORTS.

8.9 TEST 7 - TEST UNLOAD COMMAND THROUGH PORT 'B'

VERIFY THAT THE UNLOAD COMMAND FUNCTIONS PROPERLY AND THAT A PORT TIMEOUT WILL NOT OCCUR WHILE THE 'GO' BIT IS SET.

- A. ISSUE AN UNLOAD COMMAND THROUGH PORT 'B'; VERIFY THAT THE DRIVE IS SEIZED.
- B. WAIT THE MEASURED TIMEOUT INTERVAL + 25%; VERIFY THAT THE DRIVE DOES NOT TIME OUT. VERIFY THAT THE 'GO' BIT IS STILL SET AND THAT 'DRY' AND 'PIP' ARE NOT SET.
- C. REQUEST THAT THE OPERATOR PRESS THE 'STANDBY' BUTTON ON THE DRIVE.
- D. WHEN THE DRIVE CYCLES UP, VERIFY THAT THE DRIVE IS STILL SEIZED BY PORT B, THAT THE 'VV' BIT FOR PORT B IS RESET, THAT THE ATTENTION BIT FOR PORT 'B' IS SET, AND THAT THE ATTENTION BIT FOR PORT 'A' IS NOT SET.
- E. WAIT FOR THE PORT TIMEOUT TO RELEASE THE DRIVE. WHEN THE TIMEOUT OCCURS, VERIFY THAT THE DRIVE RETURNED TO NEUTRAL, THAT THE ATTENTION BIT FOR PORT 'B' IS STILL SET, AND THAT THE ATTENTION BIT FOR PORT 'A' IS NOT SET.
- F. VERIFY THAT THE 'VV' BIT FOR PORT 'A' IS NOT SET.
- G. ISSUE A PACK ACKNOWLEDGE INSTRUCTION THROUGH BOTH PORTS.

8.10 TEST 10 - TEST THE CONTROLLER SELECT SWITCH THROUGH PORT 'A' DURING UNLOAD

VERIFY THAT THE 'CONTROLLER SELECT' SWITCH IS INHIBITED WHEN THE RPO4 IS CYCLED DOWN BY AN UNLOAD COMMAND.

- A. ISSUE AN UNLOAD COMMAND THROUGH PORT 'A'
- B. REQUEST THAT THE OPERATOR SWITCH THE 'CONTROLLER SELECT' SWITCH TO 'A' AND THEN PRESS THE 'STANDBY' BUTTON.
- C. WAIT FOR 'MOL' TO SET BY MONITORING RPO51 THROUGH PORT 'A'. WHEN THE DRIVE HAS CYCLED UP ('MOL' HAS SET), ISSUE A RELEASE COMMAND THROUGH PORT 'A'.



588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
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

- D. VERIFY THAT THE DRIVE CAN BE ACCESSED BY BOTH PORTS AND THAT THE DRIVE IS STILL IN NEUTRAL.
- E. REQUEST THAT THE OPERATOR RETURN THE 'CONTROLLER SELECT' SWITCH TO 'A/B'.

8.11 TEST 11 - TEST 'CONTROLLER SELECT' SWITCH THROUGH PORT 'B'  
DURING UNLOAD

VERIFY THAT THE 'CONTROLLER SELECT' SWITCH IS INHIBITED WHEN THE RPO4 IS CYCLED DOWN BY AN UNLOAD COMMAND.

- A. ISSUE AN UNLOAD COMMAND THROUGH PORT 'B'.
- B. REQUEST THAT THE OPERATOR SWITCH THE 'CONTROLLER SELECT' SWITCH TO 'B' AND THEN PRESS THE 'STANDBY' BUTTON.
- C. WAIT FOR 'MOL' TO SET BY MONITORING RPO51 THROUGH PORT 'B'. WHEN THE DRIVE HAS CYCLED UP ('MOL' HAS SET), ISSUE A RELEASE COMMAND THROUGH PORT 'B'.
- D. VERIFY THAT THE DRIVE CAN BE ACCESSED BY BOTH PORTS AND THAT THE DRIVE IS STILL IN NEUTRAL.
- E. REQUEST THAT THE OPERATOR RETURN THE 'CONTROLLER SELECT' SWITCH TO 'A/B'.

8.12 TEST 12 - TEST 'CONTROLLER SELECT' SWITCH, DRIVE CYCLED UP

TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED UP).

- A. SWITCH TO CONTROLLER 'A' POSITION. VERIFY THAT THE DRIVE IS IN NEUTRAL AND THAT THE STATUS BITS IN RPO51, AS READ THROUGH BOTH PORTS, ARE CORRECT.
- B. SWITCH TO CONTROLLER 'B' POSITION. VERIFY THAT THE DRIVE IS IN NEUTRAL AND THAT THE STATUS BITS IN RPO51, AS READ THROUGH BOTH PORTS, ARE CORRECT.

C. RETURN THE 'CONTROLLER SELECT' SWITCH LOCKED ON PORT 'A'

TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED DOWN).

- A. CYCLE THE DRIVE DOWN.
- B. SWITCH TO CONTROLLER 'A' POSITION. VERIFY THAT THE DRIVE IS IN NEUTRAL AND THAT THE STATUS BITS IN RPO51, AS READ THROUGH BOTH PORTS, ARE CORRECT.
- C. SWITCH THE 'CONTROLLER SELECT' SWITCH TO 'A'; CYCLE THE DRIVE UP.
- D. WHEN THE DRIVE CYCLES UP, VERIFY THAT 'VV-A' IS RESET, AND THAT 'ATA-A' IS SET.
- E. ISSUE A DRIVE CLEAR COMMAND AND A READIN PRESET COMMAND THROUGH PORT 'A'.

644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
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

F. VERIFY THAT THE DRIVE CANNOT BE ACCESSED THROUGH PORT 'B' AND 'NED' DOES NOT SET WHEN ATTEMPTING TO ACCESS THE DRIVE THROUGH PORT 'B' ATTEMPT TO SET PORT REQUEST BY WRITING 0'S INTO RPDS1 THROUGH PORT 'B'.

G. ISSUE A RELEASE COMMAND THROUGH PORT 'A'. VERIFY THAT THE DRIVE REMAINS LOCKED ON PORT 'A'.

8.14 TEST 14 - TEST 'CONTROLLER SELECT' SWITCH, LOCKED ON PORT 'B'  
TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED DOWN).

A. CYCLE THE DRIVE DOWN.

B. SWITCH TO CONTROLLER 'B' POSITION. VERIFY THAT THE DRIVE IS IN NEUTRAL AND THAT THE STATUS BITS IN RPDS1, AS READ THROUGH BOTH PORTS, ARE CORRECT.

C. SWITCH THE 'CONTROLLER SELECT' SWITCH TO 'B'; CYCLE THE DRIVE UP.

D. WHEN THE DRIVE CYCLES UP, VERIFY THAT 'VV-B' IS RESET, AND THAT 'ATA-B' IS SET.

E. ISSUE A DRIVE CLEAR COMMAND AND A READIN PRESET COMMAND THROUGH PORT 'B'.

F. VERIFY THAT THE DRIVE CANNOT BE ACCESSED THROUGH PORT 'A' AND 'NED' DOES NOT SET WHEN ATTEMPTING TO ACCESS THE DRIVE THROUGH PORT 'A'. ATTEMPT TO SET PORT REQUEST BY WRITING 0'S INTO RPDS1 THROUGH PORT 'A'.

G. ISSUE A RELEASE COMMAND THROUGH PORT 'B'. VERIFY THAT THE DRIVE REMAINS LOCKED ON PORT 'B'.

H. CYCLE THE DRIVE DOWN. CHANGE THE 'CONTROLLER SELECT' TO 'A/B'; CYCLE THE DRIVE UP.

I. VERIFY THAT BOTH PORTS CAN ACCESS THE DRIVE, THAT BOTH ATTENTION BITS ARE SET, AND THAT BOTH 'VV' BITS ARE RESET.

9. PROGRAM LISTING

2

.TITLE MD-11-DZRJF-A, RPO4/5/6 DUAL CONTRLR LOGIC TEST - PART 2  
: \*COPYRIGHT (C) 1976  
: \*DIGITAL EQUIPMENT CORP.  
: \*MAYNARD, MASS. 01754  
: \*  
: \*PROGRAM BY C. MESS  
: \*

700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
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

001100  
  
  
  
000011  
000012  
000015  
000200  
177776  
  
177774  
177772  
177570  
177570  
  
  
000000  
000001  
000002  
000003  
000004  
000005  
000006  
000007  
000006  
000007  
  
  
000000  
000040  
000100  
000140  
000200  
000240  
000300  
000340

```

;#THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;#PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
;#
.SBTTL OPERATIONAL SWITCH SETTINGS
;#
;# SWITCH USE
;# -----
;# 15 HALT ON ERROR
;# 14 LOOP ON TEST
;# 13 INHIBIT ERROR TYPEOUTS
;# 11 INHIBIT ITERATIONS
;# 10 BELL ON ERROR
;# 9 LOOP ON ERROR

.SBTTL BASIC DEFINITIONS
;#INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV ENT.ERROR ;:BASIC DEFINITION OF ERROR CALL
.EQUIV IOT.SCOPE ;:BASIC DEFINITION OF SCOPE CALL

;#MISCELLANEOUS DEFINITIONS
HT= 11 ;:CODE FOR HORIZONTAL TAB
LF= 12 ;:CODE FOR LINE FEED
CR= 15 ;:CODE FOR CARRIAGE RETURN
CRLF= 200 ;:CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776 ;:PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLM= 177774 ;:STACK LIMIT REGISTER
PIRQ= 177772 ;:PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570 ;:HARDWARE SWITCH REGISTER
DOISP= 177570 ;:HARDWARE DISPLAY REGISTER

;#GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0 ;:GENERAL REGISTER
R1= %1 ;:GENERAL REGISTER
R2= %2 ;:GENERAL REGISTER
R3= %3 ;:GENERAL REGISTER
R4= %4 ;:GENERAL REGISTER
R5= %5 ;:GENERAL REGISTER
R6= %6 ;:GENERAL REGISTER
R7= %7 ;:GENERAL REGISTER
SP= %6 ;:STACK POINTER
PC= %7 ;:PROGRAM COUNTER

;#PRIORITY LEVEL DEFINITIONS
PR0= 0 ;:PRIORITY LEVEL 0
PR1= 40 ;:PRIORITY LEVEL 1
PR2= 100 ;:PRIORITY LEVEL 2
PR3= 140 ;:PRIORITY LEVEL 3
PR4= 200 ;:PRIORITY LEVEL 4
PR5= 240 ;:PRIORITY LEVEL 5
PR6= 300 ;:PRIORITY LEVEL 6
PR7= 340 ;:PRIORITY LEVEL 7
    
```

756  
757  
758 100000  
759 040000  
760 020000  
761 010000  
762 004000  
763 002000  
764 001000  
765 000400  
766 000200  
767 000100  
768 000040  
769 000020  
770 000010  
771 000004  
772 000002  
773 000001  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786 100000  
787 040000  
788 020000  
789 010000  
790 004000  
791 002000  
792 001000  
793 000400  
794 000200  
795 000100  
796 000040  
797 000020  
798 000010  
799 000004  
800 000002  
801 000001  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811

.\*"SWITCH REGISTER" SWITCH DEFINITIONS

SW15= 100000  
SW14= 40000  
SW13= 20000  
SW12= 10000  
SW11= 4000  
SW10= 2000  
SW09= 1000  
SW08= 400  
SW07= 200  
SW06= 100  
SW05= 40  
SW04= 20  
SW03= 10  
SW02= 4  
SW01= 2  
SW00= 1  
.EQUIV SW09,SW9  
.EQUIV SW08,SW8  
.EQUIV SW07,SW7  
.EQUIV SW06,SW6  
.EQUIV SW05,SW5  
.EQUIV SW04,SW4  
.EQUIV SW03,SW3  
.EQUIV SW02,SW2  
.EQUIV SW01,SW1  
.EQUIV SW00,SW0

.\*DATA BIT DEFINITIONS (BIT00 TO BIT15)

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



```

812
813
814      000004      ;#BASIC "CPU" TRAP VECTOR ADDRESSES
815      000010      ERRVEC= 4      ;: TIME OUT AND OTHER ERRORS
816      000014      RESVEC= 10     ;: RESERVED AND ILLEGAL INSTRUCTIONS
817      000014      TBITVEC=14     ;: "T" BIT
818      000014      TRIVEC= 14     ;: TRACE TRAP
819      000020      BPTVEC= 14     ;: BREAKPOINT TRAP (BPT)
820      000024      IOTVEC= 20     ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
821      000030      PWRVEC= 24     ;: POWER FAIL
822      000034      EMTVEC= 30     ;: EMULATOR TRAP (EMT) **ERROR**
823      000034      TRAPVEC=34     ;: "TRAP" TRAP
824      000060      TKVEC= 60      ;: TTY KEYBOARD VECTOR
825      000064      TPVEC= 64      ;: TTY PRINTER VECTOR
826      000240      PIRQVEC=240    ;: PROGRAM INTERRUPT REQUEST VECTOR
827
828      ;:*****
829      .SBTTL RH11 REGISTERS
830
831      ;:*****
832      ;CONTROL AND STATUS REGISTER 1 (RPCS1)
833
834      000100      IE= 100          ;: INTERRUPT ENABLE (BIT #6)
835      000200      RDY= 200        ;: READY (BIT #7)
836      000400      A16= 400        ;: HIGH ORDER BUS ADDRESS BIT (BIT #8)
837      001000      A17= 1000       ;: HIGH ORDER BUS ADDRESS BIT (BIT #9)
838      002000      PSEL= 2000      ;: PORT SELECT (BIT #10)
839      020000      MCPE= 20000     ;: MASSBUS PARITY ERROR (BIT #13)
840      040000      TRE= 40000      ;: TRANSFER ERROR (BIT #14)
841      100000      SC= 100000     ;: SPECIAL CONDITION (BIT #15)
842
843      ;WORD COUNT REGISTER (RPWC)
844      ;(EACH BIT IS CALLED BY BIT NUMBER)
845
846      ;BUS ADDRESS REGISTER (RPBA)
847      ;(EACH BIT IS CALLED BY BIT NUMBER)
848
849      ;CONTROL AND STATUS REGISTER 2 (RPCS2)
850
851      000001      US1= 1           ;: UNIT SELECT (BIT #0)
852      000002      US2= 2           ;: UNIT SELECT (BIT #1)
853      000004      US4= 4           ;: UNIT SELECT (BIT #2)
854      000010      BAI= 10          ;: BUS ADDRESS INCREMENT INHIBIT (BIT #3)
855      000020      PAT= 20          ;: MASSBUS PARITY TEST (BIT #4)
856      000040      CLR= 40          ;: CLEAR (BIT #5)
857      000100      IR= 100          ;: INPUT READY (BIT #6)
858      000200      OR= 200          ;: OUTPUT READY (BIT #7)
859      000400      MPE= 400         ;: MASS BUS PARITY ERROR (BIT #8)
860      001000      MXF= 1000        ;: MISSED TRANSFER ERROR (BIT #9)
861      002000      PGE= 2000        ;: PROGRAM ERROR (BIT #10)
862      004000      NEM= 4000        ;: NON EXISTENT MEMORY (BIT #11)
863      010000      NED= 10000       ;: NON EXISTENT DRIVE (BIT #12)
864      020000      LPE= 20000       ;: UNIBUS PARITY ERROR (BIT #13)
865      040000      WCE= 40000       ;: WRITE CHECK ERROR (BIT #14)
866      100000      DLT= 100000      ;: DATA LATE (BIT #15)
867
    
```

;DATA BUFFER REGISTER (RPDB)  
;(EACH BIT IS CALLED BY BIT NUMBER)

;\*\*\*\*\*

.SBTTL RP04/5/6 REGISTERS

;\*\*\*\*\*

;CONTROL AND STATUS 1 REGISTER. (#00)

868				
869				
870				
871				
872				
873				
874				
875				
876				
877				
878				
879				
880				
881	000001	GO=	1	;GO BIT (BIT #0)
882	000002	F1=	2	;FUNCTION CODE BIT #1
883	000004	F2=	4	;FUNCTION CODE BIT #2
884	000010	F3=	10	;FUNCTION CODE BIT #3
885	000020	F4=	20	;FUNCTION CODE BIT #4
886	000040	F5=	40	;FUNCTION CODE BIT #5
887	004000	DVA=	4000	;DEVICE AVAILABLE (BIT #11)

;DRIVE STATUS REGISTER (RPOS1) (#01)

888				
889				
890				
891				
892	000002	DFS=	1	DRIVE FORWARD 5"/SEC. (BIT #0)
893	000004	DF20=	2	DRIVE FORWARD 20"/SEC. (BIT #1)
894	000010	DIG8=	4	DRIVE TO INNER GUARD BAND (BIT #2)
895	000020	GRV=	10	GO REVERSE (BIT #3)
896	000040	DL64=	20	DIFFERENCE LESS THAN 64 (BIT #4)
897	000100	DE1=	40	DIFFERENCE EQUALS 1 (BIT #5)
898	000200	VV=	100	VOLUME VALID (BIT #6)
899	000400	DRY=	200	DRIVE READY (BIT #7)
900	001000	DFR=	400	DRIVE PRESENT (BIT #8)
901	002000	PGM=	1000	PROGRAMMABLE (BIT #9)
902	004000	LST=	2000	LAST SECTOR TRANSFERRED (BIT #10)
903	010000	WRL=	4000	WRITE LOCK (BIT #11)
904	020000	MOL=	10000	MEDIUM ON-LINE (BIT #12)
905	040000	PIP=	20000	POSITIONING OPERATION IN PROGRESS (BIT #13)
906	100000	ERR=	40000	COMPOSITE ERROR (BIT #14)
907		ATA=	100000	ATTENTION ACTIVE (BIT #15)

;ERROR REGISTER #01 (RPER1) (#02)

908				
909				
910	000001	ILF=	1	;ILLEGAL FUNCTION (BIT #0)
911	000002	ILR=	2	;ILLEGAL REGISTER (BIT #1)
912	000004	RMR=	4	;REGISTER MODIFICATION REFUSED (BIT #2)
913	000010	PAR=	10	;PARITY ERROR (BIT #3)
914	000020	FER=	20	;FORMAT ERROR (BIT #4)
915	000040	WCF=	40	;WRITE CLOCK FAIL (BIT #5)
916	000100	ECH=	100	;ECC HARD ERROR (BIT #6)
917	000200	HCE=	200	;HEADER COMPARE ERROR (BIT #7)
918	000400	HCRC=	400	;HEADER CRC ERROR (BIT #8)
919	001000	AOE=	1000	;ADDRESS OVERFLOW ERROR (BIT #9)
920	002000	IAE=	2000	;INVALID ADDRESS ERROR (BIT #10)
921	004000	WLE=	4000	;WRITE LOCK ERROR (BIT #11)
922	010000	DTE=	10000	;DRIVE TIMING ERROR (BIT #12)
923	020000	OPI=	20000	;OPERATION INCOMPLETE (BIT #13)

924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979

040000  
100000  
  
000001  
000002  
000004  
000010  
000020  
000040  
000200  
  
000001  
000002  
000004  
000010  
000020  
000040  
000100  
000200  
  
000001  
000002  
000004  
000010  
000020  
000040  
000100  
000200  
000400  
004000  
020000  
040000  
100000  
  
000001  
000002  
000004  
000010  
000020  
000040  
000100  
000200  
000400  
001000  
002000

UNS= 40000 ;DRIVE UNSAFE (BIT #14)  
DCK= 100000 ;DATA CHECK ERROR (BIT 15)  
  
;MAINTAINABILITY REGISTER (RPMR) (#03)  
DMD= 1 ;DIAGINOSTIC MODE (BIT #0)  
MCLK= 2 ;MAINTAINABILITY CLOCK (BIT #1)  
MINX= 4 ;MAINTAINABILITY INDEX (BIT #2)  
MSTCK= 10 ;MAINTAINABILITY SECTOR CLOCK (BIT #3)  
MRD= 20 ;MAINTAINABILITY READ (BIT #4)  
MWR= 40 ;MAINTAINABILITY WRITE (BIT #5)  
DTSY= 200 ;MAINTAINABILITY SYNC DETECTED (BIT #7)  
  
;ATTENTION SUMMARY PSEUDO-REGISTER (RPAS) (#04)  
AT0= 1 ;DEVICE 0 (BIT #0)  
AT1= 2 ;DEVICE 1 (BIT #1)  
AT2= 4 ;DEVICE 2 (BIT #2)  
AT3= 10 ;DEVICE 3 (BIT #3)  
AT4= 20 ;DEVICE 4 (BIT #4)  
AT5= 40 ;DEVICE 5 (BIT #5)  
AT6= 100 ;DEVICE 6 (BIT #6)  
AT7= 200 ;DEVICE 7 (BIT #7)  
  
;DESIRED SECTOR/TRACK ADDRESS REGISTER (RPDA) (#05)  
;(EACH BIT IS CALLED BY BIT NUMBER)  
  
;DRIVE TYPE REGISTER (RPDT) (#06)  
DT00= 1 ;DRIVE TYPE NUMBER BIT 1  
DT01= 2 ;DRIVE TYPE NUMBER BIT 2  
DT02= 4 ;DRIVE TYPE NUMBER BIT 3  
DT03= 10 ;DRIVE TYPE NUMBER BIT 4  
DT04= 20 ;DRIVE TYPE NUMBER BIT 5  
DT05= 40 ;DRIVE TYPE NUMBER BIT 6  
DT06= 100 ;DRIVE TYPE NUMBER BIT 7  
DT07= 200 ;DRIVE TYPE NUMBER BIT 8  
DT08= 400 ;DRIVE TYPE NUMBER BIT 9  
DRQ= 4000 ;DRIVE REQUEST REQUIRED (BIT #11)  
MOH= 20000 ;MOVING HEAD (BIT #13)  
TAP= 40000 ;TAPE DRIVE (BIT #14)  
NBA= 100000 ;NOT BLOCK ADDRESSED (BIT #15)  
  
;LOOK-AHEAD REGISTER (RPLA) (#07)  
EXT1= 1 ;EXTENSION 1 (BIT #0)  
EXT2= 2 ;EXTENSION 2 (BIT #1)  
EXT4= 4 ;EXTENSION 3 (BIT #2)  
EXT10= 10 ;EXTENSION 4 (BIT #3)  
EXT20= 20 ;EXTENSION 5 (BIT #4)  
EXT40= 40 ;EXTENSION 6 (BIT #5)  
SC1= 100 ;SECTOR COUNT FIELD 0 (BIT #6)  
SC2= 200 ;SECTOR COUNT FIELD 1 (BIT #7)  
SC4= 400 ;SECTOR COUNT FIELD 2 (BIT #8)  
SC10= 1000 ;SECTOR COUNT FIELD 3 (BIT #9)  
SC20= 2000 ;SECTOR COUNT FIELD 4 (BIT #10)

MD-11-DZRTF-A, RP04/5/6 DUAL CONTRLR LOGIC TEST - PART 2  
DZRTJFA.CMB 02-NOV-76 19:11 RP04/5/6 REGISTERS

```

980      004000      TRK1= 4000      ; TRACK FIELD 1 (BIT #11)
981      010000      TRK2= 10000     ; TRACK FIELD 2 (BIT #12)
982      020000      TRK4= 20000     ; TRACK FIELD 3 (BIT #13)
983      040000      TRK10= 40000    ; TRACK FIELD 4 (BIT #14)
984      100000      TRK20= 100000   ; TRACK FIELD 5 (BIT #15)
985
986      ;RP04 ERROR REGISTER #2 (RPER2) (#10)
987
988      000001      WCU= 1          ; WRITE CURRENT UNSAFE (BIT #0)
989      000002      CSF= 2          ; CURRENT SINK FAILURE (BIT #1)
990      000004      WSU= 4          ; WRITE SELECT UNSAFE (BIT #2)
991      000010      CSU= 10         ; CURRENT SWITCH UNSAFE (BIT #3)
992      000020      MSE= 20         ; MOTOR SEQUENCE ERROR (BIT #4)
993      000040      TDF= 40         ; TRANSITIONS DETECTOR FAILURE (BIT #5)
994      000100      TUF= 100        ; TRANSITIONS UNSAFE (BIT #6)
995      000200      FEN= 200        ; FAILSAFE ENABLED (BIT #7)
996      000400      WRU= 400        ; WRITE READY UNSAFE (BIT #8)
997      001000      MHS= 1000       ; MULTIPLE HEAD SELECT (BIT #9)
998      002000      NHS= 2000       ; NO HEAD SELECTION (BIT #10)
999      004000      IXE= 4000       ; INDEX ERROR (BIT #11)
1000     010000      VU30= 10000     ; 30VOLT UNSAFE (BIT #12)
1001     020000      PLU= 20000     ; PLO UNSAFE (BIT #13)
1002     100000      WCU= 100000    ; AC UNSAFE (BIT #15)
1003
1004     ;RP05/6 ERROR REGISTER #02 (RPER2) (#10)
1005
1006     000001      WCU= 1          ; WRITE CURRENT UNSAFE (BIT #0)
1007     000002      CSF= 2          ; CURRENT SINK FAILURE (BIT #1)
1008     000004      WSU= 4          ; WRITE SELECT UNSAFE (BIT #2)
1009     000010      CSU= 10         ; CURRENT SWITCH UNSAFE (BIT #3)
1010     000020      RAW= 20         ; READ AND WRITE (BIT #4)
1011     000040      TDF= 40         ; TRANSITIONS DETECTOR FAILURE (BIT #5)
1012     000100      TUF= 100        ; TRANSITIONS UNSAFE (BIT #6)
1013     000200      ABS= 200        ; ABNORMAL STOP (BIT #7)
1014     000400      WRU= 400        ; WRITE READY UNSAFE (BIT #8)
1015     001000      MHS= 1000       ; MULTIPLE HEAD SELECT (BIT #9)
1016     002000      NHS= 2000       ; NO HEAD SELECTION (BIT #10)
1017     004000      IXE= 4000       ; INDEX ERROR (BIT #11)
1018     020000      PLU= 20000     ; PLO UNSAFE (BIT #12)
1019
1020     ;OFFSET REGISTER (RPOF) (#11)
1021
1022     000001      OF25= 1          ; OFFSET 25 MICRO INCHES (BIT #0)
1023     000002      OF50= 2          ; OFFSET 50 MICRO INCHES (BIT #1)
1024     000004      OF100= 4         ; OFFSET 100 MICRO INCHES (BIT #2)
1025     000010      OF200= 10        ; OFFSET 200 MICRO INCHES (BIT #3)
1026     000020      OF400= 20        ; OFFSET 400 MICRO INCHES (BIT #4)
1027     000040      OF800= 40        ; OFFSET 800 MICRO INCHES (BIT #5)
1028     000200      OFREV= 200       ; OFFSET NEGATIVE (REVERSE) (BIT #5)
1029     002000      HCI= 2000       ; HEADER COMPARE INHIBIT (BIT #10)
1030     004000      ECI= 4000       ; ERROR CORRECTION CODE INHIBIT (BIT #11)
1031     010000      FMT22= 10000     ; FORMAT BIT (BIT #12)
1032
1033     ;DESIRED CYLINDER ADDRESS (RPCA) (#12)
1034     ;(EACH BIT IS CALLED BY BIT NUMBER)
1035

```



1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091

000001  
000002  
000010  
000040  
000100  
040000  
100000  
  
000001  
000002  
000040  
000100  
020000  
040000  
100000  
  
000000  
000002  
000004  
000006  
000010  
000012  
000014  
000016  
000020  
000022  
000024  
000026  
000030  
000032  
000034  
000036  
000040  
000042

;CURRENT CYLINDER ADDRESS (RPCC) (#13)  
;(EACH BIT IS CALLED BY BIT NUMBER)

;SERIAL NUMBER REGISTER (RPSN) (#14)  
;(EACH IS CALLED BY BIT NUMBER)

;RPO4 ERROR REGISTER #03 (RPER3) (#15)

PSU= 1  
VUF= 2  
UMR= 10  
ACL= 40  
DCL= 100  
SKI= 40000  
OCYL= 100000

;PACK SPEED UNSAFE (BIT #0)  
;VELOCITY UNSAFE (BIT #1)  
;ANY UNSAFE EXCEPT READ/WRITE (BIT #3)  
;AC LOW (BIT #5)  
;DC LOW (BIT #6)  
;SEEK INCOMPLETE (BIT #14)  
;OFF CYLINDER (BIT #15)

;RPO5/6 ERROR REGISTER #03 (RPER3) (#15)

DCU= 1  
WAO= 2  
ACL= 40  
DCL= 100  
OPE= 20000  
SKI= 40000  
OCYL= 100000

;DC UNSAFE (BIT #0)  
;WRITE AND OFFSET (BIT #1)  
;AC LOW (BIT #5)  
;DC LOW (BIT #6)  
;OPERATOR PLUG ERROR (BIT #13)  
;SEEK INCOMPLETE (BIT #14)  
;OFF CYLINDER ERROR (BIT #15)

;ECC POSITION REGISTER (RPEC1) (#16)  
;(EACH BIT IS CALLED BY BIT NUMBER)

;ECC PATTERN REGISTER (RPEC2) (#17)  
;(EACH BIT IS CALLED BY BIT NUMBER)

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

.SBTTL DEFINITIONS OF THE RH11/RPO4/5/6 ADDRESS INDEXES

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

RPCS1=0  
RPWC=2  
RPBA=4  
RPDA=6  
RPCS2=10  
RPDS1=12  
RPER1=14  
RPAS=16  
RPLA=20  
RPDB=22  
RPMR=24  
RPDT=26  
RPSN=30  
RPOF=32  
RPC4=34  
RPCC=36  
RPER2=40  
RPER3=42

;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)  
;WORD COUNT REGISTER (NOT A DRIVE REG.)  
;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG.)  
;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)  
;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG.)  
;DRIVE STATUS REGISTER (DRIVE REG 01)  
;ERROR REGISTER #1 (DRIVE REG. 02)  
;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)  
;LOOK AHEAD REGISTER (DRIVE REG. 07)  
;DATA BUFFER REGISTER (NOT A DRIVE REG.)  
;MAINTAINABILITY REGISTER (DRIVE REG. 03)  
;DRIVE TYPE REGISTER (DRIVE REG. 06)  
;SERIAL NUMBER REGISTER (DRIVE REG. 10)  
;OFFSET REGISTER (DRIVE REG. 11)  
;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)  
;CURRENT CYLINDER ADDRESS REGISTER (DRIVE REG. 13)  
;ERROR REGISTER #2 (DRIVE REG. 14)  
;ERROR REGISTER #3 (DRIVE REG. 15)

DEFINITIONS OF THE RH11/RPO4/5/6 ADDRESS INDEXES

```

1092          000044      RPEC1=44          ;ECC POSITION REGISTER (DRIVE REG. 16)
1093          000046      RPEC2=46          ;ECC PATTERN REGISTER (DRIVE REG. 17)
1094
1095          .SBTTL TRAP CATCHER
1096
1097          000000      .=0
1098          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1099          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1100          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1101          000174      .=174
1102 000174 000000      DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
1103 000176 000000      SWREG: .WORD 0          ;;SOFTWARE SWITCH REGISTER
1104
1105          .SBTTL ACT11 HOOKS
1106
1107          ;*****
1108          ;HOOKS REQUIRED BY ACT11
1109          000200      $SVPC=.          ;SAVE PC
1110          000046      .=46
1111 000046 017724      SENDAD          ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
1112          000052      .=52
1113 000052 020000      .WORD 20000          ;;2)SET LOC.52 TO 20000
1114          000200      .=$SVPC          ;; RESTORE PC
1115
1116          .SBTTL STARTING ADDRESS = 200
1117
1118 000200 000137 001706      JMP START          ;START THE PROGRAM
1119
1120          .SBTTL START THE PROGRAM AND CHANGE THE RH11 ADDRESS = 204
1121
1122 000204 000137 001714      JMP START1         ;START AND CHANGE THE RH11 ADDRESS
1123
1124

```

COMMON TAGS

```

1125
1126
1127
1128
1129
1130
1131 001100
1132 001100 000000
1133 001100 000000
1134 001102 000
1135 001103 000
1136 001104 000000
1137 001106 000000
1138 001110 000000
1139 001112 000000
1140 001114 000
1141 001115 001
1142 001116 000000
1143 001120 000000
1144 001122 000000
1145 001124 000000
1146 001126 000000
1147 001130 000000
1148 001132 000000
1149 001134 000
1150 001135 000
1151 001136 000000
1152 001140 177570
1153 001142 177570
1154 001144 177560
1155 001146 177562
1156 001150 177564
1157 001152 177566
1158 001154 000
1159 001155 002
1160 001156 012
1161 001157 000
1162 001160 000000
1163
1164 001162 000000
1165 001164 000000
1166 001166 000000
1167 001170 000000
1168 001172 000000
1169 001174 000000
1170 001176 000000
1171 001200 000000
1172 001202 177607 000377
1173 001206 077
1174 001207 015
1175 001210 000012
1176
1177 000015
1178 000012
1179 001212 172540
1180 001214 172542

```

.SBTTL COMMON TAGS

```

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

```

.SMTAG: =1100

```

$PASS: .WORD 0
$STNM: .BYTE 0
$ERFLG: .BYTE 0
$ICNT: .WORD 0
$LPADR: .WORD 0
$LPERR: .WORD 0
$ERTTL: .WORD 0
$ITEMB: .BYTE 0
$ERMAX: .BYTE 1
$ERRPC: .WORD 0
$GDADR: .WORD 0
$BDADR: .WORD 0
$GDADR: .WORD 0
$BDADR: .WORD 0
$AUTOB: .BYTE 0
$INTAG: .BYTE 0
$SWR: .WORD DSWR
$DISPLAY: .WORD DDISP
$TKS: 177560
$TKB: 177562
$TPS: 177564
$TPB: 177566
$NULL: .BYTE 0
$FILLS: .BYTE 2
$FILLC: .BYTE 12
$TPFLG: .BYTE 0
$REGAD: .WORD 0
$REGO: .WORD 0
$TMP0: .WORD 0
$TMP1: .WORD 0
$TMP2: .WORD 0
$TMP3: .WORD 0
$TMP4: .WORD 0
$TIMES: 0
$ESCAPE: 0
$BELL: .ASCIIZ <207><377><377>
$QUES: .ASCII /?/
$CRLF: .ASCII <15>
$LF: .ASCIIZ <12>
CR = 15
LF = 12
$LKCSR: .WORD 172540
$LKCSB: .WORD 172542

```

```

: START OF COMMON TAGS
: CONTAINS PASS COUNT
: CONTAINS THE TEST NUMBER
: CONTAINS ERROR FLAG
: CONTAINS SUBTEST ITERATION COUNT
: CONTAINS SCOPE LOOP ADDRESS
: CONTAINS SCOPE RETURN FOR ERRORS
: CONTAINS TOTAL ERRORS DETECTED
: CONTAINS ITEM CONTROL BYTE
: CONTAINS MAX. ERRORS PER TEST
: CONTAINS PC OF LAST ERROR INSTRUCTION
: CONTAINS ADDRESS OF 'GOOD' DATA
: CONTAINS ADDRESS OF 'BAD' DATA
: CONTAINS 'GOOD' DATA
: CONTAINS 'BAD' DATA
: RESERVED--NOT TO BE USED
: AUTOMATIC MODE INDICATOR
: INTERRUPT MODE INDICATOR
: ADDRESS OF SWITCH REGISTER
: ADDRESS OF DISPLAY REGISTER
: TTY KBD STATUS
: TTY KBD BUFFER
: TTY PRINTER STATUS REG. ADDRESS
: TTY PRINTER BUFFER REG. ADDRESS
: CONTAINS NULL CHARACTER FOR FILLS
: CONTAINS # OF FILLER CHARACTERS REQUIRED
: INSERT FILL CHARS. AFTER A "LINE FEED"
: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
: CONTAINS THE ADDRESS FROM WHICH ($REGO) WAS OBTAINED
: CONTAINS (($REGO)+0)
: USER DEFINED
: USER DEFINED
: USER DEFINED
: USER DEFINED
: USER DEFINED
: MAX. NUMBER OF ITERATIONS
: ESCAPE ON ERROR ADDRESS
: CODE FOR BELL
: QUESTION MARK
: CARRIAGE RETURN
: LINE FEED
: ADDR OF KW11-P STATUS REGISTER
: ADDR OF KW11-P COUNTER BUFFER

```

K02

1181	001216	000104	\$L'VEC: .WORD	104	; ADDR OF KW11-P VECTOR
1182	001220	177546	\$LKS: .WORD	177546	; ADDR OF KW11-L STATUS REGISTER
1183	001222	000100	\$LLVEC: .WORD	100	; ADDR OF KW11-L VECTOR
1184	001224	000000	PORTA: .WORD	0	; ADDRESS OF PORT A
1185	001226	000000	PORTB: .WORD	0	; ADDRESS OF PORT B
1186	001230	000000	PORTC: .WORD	0	; ADDRESS OF DIFFERENT DRIVE
1187	001232	000000	ASRI: .WORD	0	; ATA-A OR ATA-B = 1
1188	001234	000000	PTNBR: .WORD	0	; CONTAINS THE PORT ADDRESS FOR ERROR TYPEOUTS
1189	001236	000000	SEIZPT: .WORD	0	; CONTAINS THE ADDRESS OF THE SEIZING PORT
1190	001240	000000	OPPRT: .WORD	0	; CONTAINS THE ADDRESS OF THE 'OPPOSITE' PORT
1191	001242	000000	TSTNUM: .WORD	0	; NUMBER OF THE CURRENT TEST
1192	001244	000000	CKERR: .WORD	0	; IF -1, A REGISTER MISCOMPARISON OCCURRED
1193	001246	000000	NOSEIZ: .WORD	0	; IF -1, THE PORT IN 'SEIZPT' DID NOT SEIZE THE DRIVE
1194	001250	000000	RELERR: .WORD	0	; IF -1, THE PORT IN 'SEIZPT' DID NOT RELEASE THE DRIVE
1195	001252	000000	TIME: .WORD	0	; ELAPSED TIME COUNTER
1196	001254	000000	WATCH: .WORD	0	; WATCH DOG TIMER LOCATION
1197	001256	000000	TIMEA: .WORD	0	; THE TIMEOUT ONE-SHOT VALUE MEASURED THROUGH PORT A
1198	001260	000000	TIMEAP: .WORD	0	; PORT A TIMEOUT VALUE + 25%
1199	001262	000000	TIMEB: .WORD	0	; THE TIMEOUT ONE-SHOT VALUE MEASURED THROUGH PORT B
1200	001264	000000	TIMEBP: .WORD	0	; PORT B TIMEOUT VALUE + 25%
1201	001266	000000	KYBCTL: .WORD	0	; SINGLE TEST INDICATOR
1202	001270	000000	CHGADR: .WORD	0	; CHANGE THE R'11 ADDRESS INDICATOR
1203					
1204					
1205					
1206					
1207					
1208					
1209					
1210	001272	176700	\$RPAOR: .WORD	176700	; RH11/RPO4/5/6 UNIBUS ADDRESS
1211	001274	000254	\$RPVEC: .WORD	254	; RH11 INTERRUPT VECTOR ADDRESS
1212					



ERROR POINTER TABLE

1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268

.SBTTL ERROR POINTER TABLE

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

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

001276

\$ERRTB:

;ERROR 1

001276 024457 EM1 ;DRIVE IS NON-EXISTENT ('MED' BIT SET)  
001300 027227 DH1  
001302 030610 DT1  
001304 031030 DF1

;ERROR 2

001306 024525 EM2 ;WRONG DRIVE TYPE  
001310 027300 DH2  
001312 030624 DT2  
001314 031035 DF2

;ERROR 3

001316 024546 EM3 ;CONTROLLER SELECT SWITCH ON DRIVE NOT IN 'A/B'  
001320 027227 DH1  
001322 030610 DT1  
001324 031030 DF1

;ERROR 4

001326 024625 EM4 ;DRIVE NOT ON LINE  
001330 027300 DH2  
001332 030624 DT2  
001334 031035 DF2

;ERROR 5

001336 024647 EMS ;SERIAL NUMBER READ THROUGH EACH PORT NOT THE SAME  
001340 027354 DHS  
001342 030642 DTS  
001344 031043 DFS

;ERROR 6

001346 024731 EM6 ;TIMEOUT HAS NOT OCCURRED WITHIN 2 SECONDS  
001350 027423 DHS

MD-11-DZRJF-A, RP04/5/6 DUAL CONTRLR LOGIC TEST - PART 2  
 DZRJFA.CMB 02-NOV-76 19:11 ERROR POINTER TABLE

1269	001352	030656	DT6	
1270	001354	031050	DF6	
1271				
1272				;ERROR 7
1273				
1274	001356	025003	EM7	;TIMEOUT ONE-SHOT IS LESS THAN 500 MS
1275	001360	027452	DH7	
1276	001362	030666	DT7	
1277	001364	031053	DF7	
1278				
1279				;ERROR 10
1280				
1281	001366	025050	EM10	;READIN PRESET DOES NOT SET VOLUME VALID FOR THE PORT
1282	001370	027227	DH1	
1283	001372	030610	DT1	
1284	001374	031030	DF1	
1285				
1286				;ERROR 11
1287				
1288	001376	025135	EM11	; 'GO' BIT RESET DURING UNLOAD COMMAND
1289	001400	027227	DH1	
1290	001402	030610	DT1	
1291	001404	031030	DF1	
1292				
1293				;ERROR 12
1294				
1295	001406	025202	EM12	; INCORRECT STATUS DURING UNLOAD COMMAND
1296	001410	027227	DH1	
1297	001412	030610	DT1	
1298	001414	031030	DF1	
1299				
1300				;ERROR 13
1301				
1302	001416	025251	EM13	;DRIVE DID NOT RETURN TO NEUTRAL AFTER UNLOAD COMMAND
1303	001420	027517	DH13	
1304	001422	030700	DT13	
1305	001424	031050	DF6	
1306				
1307				;ERROR 14
1308				
1309	001426	025336	EM14	;ATTENTION BIT SET ON 'OPPOSITE PORT' AFTER UNLOAD
1310	001430	027575	DH14	
1311	001432	030710	DT14	
1312	001434	031057	DF14	
1313				
1314				;ERROR 15
1315				
1316	001436	025420	EM15	;ATTENTION BIT NOT SET ON PORT WHICH ISSUED 'UNLOAD'
1317	001440	027227	DH1	
1318	001442	030610	DT1	
1319	001444	031030	DF1	
1320				
1321				;ERROR 16
1322				
1323	001446	025504	EM16	;DRIVE NOT IN NEUTRAL AFTER UNLOAD WITH 'CONTROLLER ;SELECT' SWITCH MOVED FROM 'A/B'
1324				

1325	001450	027517	DH13	
1326	001452	030700	DT13	
1327	001454	031050	DF6	
1328				;ERROR 17
1329				
1330	001456	025630	EM17	;DRIVE LOCKED ON PORT 'A' BY SWITCH WHILE CYCLED UP
1331	001460	027715	DH17	
1332	001462	030726	DT17	
1333	001464	031065	DF17	
1334				
1335				;ERROR 20
1336				
1337	001466	025713	EM20	;DRIVE LOCKED ON PORT 'B' BY SWITCH WHILE CYCLED UP
1338	001470	027715	DH17	
1339	001472	030726	DT17	
1340	001474	031065	DF17	
1341				
1342				;ERROR 21
1343				
1344	001476	025776	EM21	;STATUS INCORRECT FOR PORT AFTER CYCLE UP
1345	001500	027227	DH1	
1346	001502	030610	DT1	
1347	001504	031030	DF1	
1348				
1349				;ERROR 22
1350				
1351	001506	026047	EM22	;REGISTER CONTENTS SEEN WHEN DRIVE SWITCHED ON 'OPPOSITE' PORT
1352	001510	027227	DH1	
1353	001512	030610	DT1	
1354	001514	031030	DF1	
1355				
1356				;ERROR 23
1357				
1358	001516	026145	EM23	; 'NED' SET WHEN RPO51 ACCESSED THROUGH PORT NOT SWITCHED
1359	001520	027227	DH1	
1360	001522	030610	DT1	
1361	001524	031030	DF1	
1362				
1363				;ERROR 24
1364				
1365	001526	026235	EM24	;DRIVE SWITCHED TO LOCKED OUT PORT WHEN RELEASED/
1366	001530	027734	DH24	
1367	001532	030734	DT24	
1368	001534	031053	DF7	
1369				
1370				;ERROR 25
1371				
1372	001536	026315	EM25	;RH11 DIDN'T RESPOND TO ADDRESSING
1373	001540	030040	DH25	
1374	001542	030746	DT25	
1375	001544	031067	DF25	
1376				
1377				;ERROR 26
1378				
1379	001546	000000	0	;UNUSED ERROR MESSAGES
1380	001550	000000	0	

1381	001552	000000	0	
1382	001554	000000	0	
1383				
1384				;ERROR 27
1385				
1386	001556	000000	0	;UNUSED ERROR MESSAGES
1387	001560	000000	0	
1388	001562	000000	0	
1389	001564	000000	0	
1390				
1391				;ERROR 30
1392				
1393	001566	026357	EM30	;DRIVE NOT SEIZED BY PORT 'N'
1394	001570	030047	DH30	
1395	001572	030752	DT30	
1396	001574	031070	DF30	
1397				
1398				;ERROR 31
1399				
1400	001576	026410	EM31	;WRONG STATUS SEEN BY THE SEIZING PORT
1401	001600	027300	DH2	
1402	001602	030624	DT2	
1403	001604	031035	DF2	
1404				
1405				;ERROR 32
1406				
1407	001606	026456	EM32	;REGISTER CONTENTS INCORRECT
1408	001610	027300	DH2	
1409	001612	030624	DT2	
1410	001614	031035	DF2	
1411				
1412				;ERROR 33
1413				
1414	001616	026506	EM33	;CONTROL BUS PARITY ERROR WHILE READING REGISTER
1415	001620	027227	DF1	
1416	001622	030619	DT1	
1417	001624	031030	DF1	
1418				
1419				;ERROR 34
1420				
1421	001626	026572	EM34	;CAN'T ACCESS DRIVE THROUGH EITHER PORT
1422	001630	030172	DH34	
1423	001632	030772	DT34	
1424	001634	031077	DF34	
1425				
1426				;ERROR 35
1427				
1428	001636	026641	EM35	;DRIVE NOT IN NEUTRAL AFTER RELEASE, REQUEST NOT SET
1429	001640	030270	DH35	
1430	001642	031004	DT35	
1431	001644	031053	DF7	
1432				
1433				;ERROR 36
1434				
1435	001646	026726	EM36	;DRIVE NOT IN NEUTRAL AFTER TIMEOUT, REQUEST NOT SET
1436	001650	030270	DH35	

1437	001652	031004			DT35	
1438	001654	031053			DF7	
1439						
1440						;ERROR 37
1441						
1442	001656	027013			EM37	;REGISTER CONTENTS INCORRECT AFTER RELEASE/TIMEOUT
1443	001660	030366			DH37	
1444	001662	030752			DT30	
1445	001664	031070			DF30	
1446						
1447						;ERROR 40
1448						
1449	001666	027074			EM40	;DRIVE NOT SEIZED BY PORT AFTER RELEASE WITH REQUEST SET
1450	001670	030511			DH40	
1451	001672	031016			DT40	
1452	001674	031053			DF7	
1453						
1454						;ERROR 41
1455						
1456	001676	027151			EM41	;REGISTER WRONG AFTER RELEASE WITH REQUEST SET
1457	001700	030047			DH30	
1458	001702	030752			DT30	
1459	001704	031070			DF30	
1460						
1461						
1462						
1463						
1464						;*****
1465						
1466						.SBTTL STARTUP AND INITIALIZATION ROUTINES
1467						
1468						;*****
1469						
1470	001706	005037	001270		START: CLR CHGADR	;CLEAR THE 'CHANGE RH11 ADDRESS' INDICATOR
1471	001712	000403			BR START2	;GO TO THE START
1472	001714	012737	177777	001270	START1: MOV @-1,CHGADR	;SET THE 'CHANGE RH11 ADDRESS' INDICATOR
1473	001722	000005			START2: RESET	;CLEAR THE BUS
1474					.SBTTL INITIALIZE THE COMMON TAGS	
1475					;CLEAR THE COMMON TAGS (SCHTAG) AREA	
1476	001724	012706	001100		MOV #SCHTAG,R6	;FIRST LOCATION TO BE CLEARED
1477	001730	005026			CLR (R6)+	;CLEAR MEMORY LOCATION
1478	001732	022706	001140		CMP #SWR,R6 ;;DONE?	
1479	001736	001374			BNE .-6	;LOOP BACK IF NO
1480	001740	012706	001100		MOV #STACK,SP	;SETUP THE STACK POINTER
1481					;INITIALIZE A FEW VECTORS	
1482	001744	012737	020156	000020	MOV #SCOPE,@IOTVEC	;IOT VECTOR FOR SCOPE ROUTINE
1483	001752	012737	000340	000022	MOV #340,@IOTVEC+2	;LEVEL 7
1484	001760	012737	020340	000030	MOV #SERADR,@EMTVEC	;EMT VECTOR FOR ERROR ROUTINE
1485	001766	012737	000340	000032	MOV #340,@EMTVEC+2	;LEVEL 7
1486	001774	012737	023140	000034	MOV #STRAP,@TRAPVEC	;TRAP VECTOR FOR TRAP CALLS
1487	002002	012737	000340	000036	MOV #340,@TRAPVEC+2	;LEVEL 7
1488	002010	013737	017570	017562	MOV SENDCT,SEOPCT	;SETUP END-OF-PROGRAM COUNTER
1489	002016	005037	001176		CLR \$TIMES	;INITIALIZE NUMBER OF ITERATIONS
1490	002022	005037	001200		CLR \$ESCAPE	;CLEAR THE ESCAPE ON ERROR ADDRESS
1491	002026	012737	000001	001115	MOV #1,\$ERMAX	;ALLOW ONE ERROR PER TEST
1492	002034	012737	002034	001106	MOV #.,\$LPADR	;INITIALIZE THE LOOP ADDRESS FOR SCOPE

```

1493 002042 012737 002042 001110      MOV      #,SLPERR      ;; SETUP THE ERROR LOOP ADDRESS
1494                                     ;; SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1495                                     ;; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
1496 002050 013746 000004      MOV      @ERRVEC, -(SP) ;; SAVE ERROR VECTOR
1497 002054 012737 002110 000004      MOV      @64$, @ERRVEC ;; SET UP ERROR VECTOR
1498 002062 012737 177570 001140      MOV      @OSWR, SWR    ;; SETUP FOR A HARDWARE SWICH REGISTER
1499 002070 012737 177570 001142      MOV      @DISP, DISPLAY ;; AND A HARDWARE DISPLAY REGISTER
1500 002076 022777 177777 177034      CMP      #-1, @SWR    ;; TRY TO REFERENCE HARDWARE SWR
1501 002104 001012      BNE     66$          ;; BRANCH IF NO TIMEOUT TRAP OCCURRED
1502                                     ;; AND THE HARDWARE SWR IS NOT = -1
1503 002106 000403      BR      65$        ;; BRANCH IF NO TIMEOUT
1504 002110 012716 002116      64$: MOV      @65$, (SP) ;; SET UP FOR TRAP RETURN
1505 002114 000002      RTI
1506 002116 012737 000176 001140 65$: MOV      @SWREG, SWR  ;; POINT TO SOFTWARE SWR
1507 002124 012737 000174 001142      MOV      @DISPREG, DISPLAY
1508 002132 012637 000004 66$: MOV      (SP)+, @ERRVEC ;; RESTORE ERROR VECTOR
1509
1510 002136 005227 177777      INC      #-1        ;; FIRST START ?
1511 002142 001002      BNE     1$         ;; BR IF NOT
1512 002144 104401 023226      TYPE    TITLE      ;; TYPE PROGRAM NAME
1513 002150 004737 021550      1$: JSR     PC, $TKINT ;; SETUP THE TTY KEYBOARD
1514                                     .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
1515 002154 005737 000042      TST     @042       ;; ARE WE RUNNING UNDER XXDP/ACT?
1516 002160 001006      BNE     67$        ;; BRANCH IF YES
1517 002162 023727 001140 000176      CMP     SWR, @SWREG ;; SOFTWARE SWITCH REG SELECTED?
1518 002170 001005      BNE     68$        ;; BRANCH IF NO
1519 002172 104406      GTSWR          ;; GET SOFT-SWR SETTINGS
1520 002174 000403      BR      68$
1521 002176 112737 000001 001134 67$: MOV     @1, $AUTOB  ;; SET AUTO-MODE INDICATOR
1522 002204      68$:
1523 002204 004737 002576      JSR     PC, CHANGE  ;; CHECK/CHANGE THE RH11 ADDRESS
1524 002210 104401 023335      TYPE    ,ENTERA    ;; ENTER DRIVE ADDRESS
1525 002214 104412      RDOCT          ;; GET THE ADDRESS
1526 002216 012637 001224      MOV     (SP)+, PORTA ;; STORE THE ADDRESS
1527 002222 023727 001224 000007      CMP     PORTA, #7   ;; SEE IF ADDRESS TOO LARGE
1528 002230 101403      BLOS    2$         ;; BR IF NOT
1529 002232 104401 023365      TYPE    ,ADRERR    ;; TYPE ADDRESS ERROR MESSAGE
1530 002236 000744      BR      1$         ;; TRY AGAIN
1531 002240 013737 001224 001226 2$: MOV     PORTA, PORTB ;; GENERATE THE PORT B ADDRESS
1532 002246 005237 001226      INC     PORTB      ;; INCREMENT THE ADDRESS
1533 002252 042737 000016 001226      BIC     #16, PORTB ;; LEAVE BIT 0
1534 002260 013746 001224      MOV     PORTA, -(SP) ;; PUT PORT A ADDRESS ON THE STACK
1535 002264 042716 177771      BIC     #16, (SP)  ;; SAVE BITS 1 & 2
1536 002270 052637 001226      BIS     (SP)+, PORTB ;; SET BITS 1 & 2 IN PORT B ADDRESS
1537 002274 104401 023407      TYPE    ,PORTAIS   ;; 'PORT A ADDRESS IS '
1538 002300 013746 001224      MOV     PORTA, -(SP) ;; SAVE PORTA FOR TYPEOUT
1539                                     ;; TYPE PORT A ADDRESS
1540 002304 104403      TYPOS          ;; GO TYPE--OCTAL ASCII
1541 002306      .BYTE 1          ;; TYPE 1 DIGIT(S)
1542 002307      .BYTE 0          ;; SUPPRESS LEADING ZEROS
1543 002310 104401 023435      TYPE    ,PORTBIS   ;; 'PORT B ADDRESS IS '
1544 002314 013746 001226      MOV     PORTB, -(SP) ;; SAVE PORTB FOR TYPEOUT
1545                                     ;; TYPE PORT B ADDRESS
1546 002320 104403      TYPOS          ;; GO TYPE--OCTAL ASCII
1547 002322      .BYTE 1          ;; TYPE 1 DIGIT(S)
1548 002323      .BYTE 0          ;; SUPPRESS LEADING ZEROS

```



# E03

```

1549 002324 104401 001207          TYPE      $CRLF          ; ANOTHER CR-LF
1550 002330 013737 001224 001230      MOV      PORTA,PORTC ; GENERATE ADDRESS OF DRIVE NOT TESTED
1551 002336 062737 000006 001230      ADD      #6,PORTC   ; COMPLEMENT SOME BITS
1552 002344 042737 177770 001230      BIC      #1C7,PORTC ; SAVE ONLY LOWER BITS
1553 002352 013701 001224          MOV      PORTA,R1   ; USE PORT A ADDRESS AS INDEX
1554 002356 116137 031134 001232      MOVB    ATABIT(R1),ASR1 ; GET ATTENTION BIT FOR DRIVE
1555 002364 005037 001256          CLR      TIMEA     ; CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1556 002370 005037 001260          CLR      TIMEAP    ; CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1557 002374 005037 001262          CLR      TIMEB     ; CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1558 002400 005037 001264          CLR      TIMEBP    ; CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1559 002404 004737 017744          JSR      PC,CKCLK  ; SETUP CLOCK
1560 002410 000137 002424          JMP      EXEC      ; CLOCK HAS BEEN STARTED
1561 002414 104401 023463          TYPE      ,NOCLOCK ; NO CLOCK ON SYSTEM
1562 002420 000000          HALT     ; FATAL ERROR
1563 002422 000776          BR      3$        ; INTERLOCK THE HALT
1564
1565
1566          ;ROUTINE TO GET THE TEST NUMBER FROM THE OPERATOR
1567          EXEC:   RESET          ; CLEAR EVERYTHING
1568          CLR      PS           ; CLEAR THE PROCESSOR STATUS WORD
1569          TYPE      $CRLF      ; CR-LF
1570          MOV      $RPNADR,RO  ; RH11 ADDRESS FOR INDEXING
1571          MOV      #STACK,SP   ; LOAD STACK POINTER
1572          JSR      PC,CKCLK    ; START THE CLOCK
1573          NOP                ; RETURN IF NO CLOCK
1574          JSR      PC,$TKINT   ; INITIALIZE THE KEYBOARD
1575          CLR      KYBCTL      ; CLEAR SINGLE TEST INDICATOR
1576          CLR      $PASS      ; CLEAR THE PASS COUNT
1577          MOVB    #1,$ERMAX    ; SET ERROR MAX TO 1
1578          MOV      #,$SLPNADR  ; INITIAL SETTING FOR LOOP ADDRESS
1579          MOV      #,$SLPERR   ; INITIAL SETTING FOR LOOP ON ERROR ADDRESS
1580          TYPE      ,TESTNO   ; ASK FOR TEST NUMBER
1581          RDOCT          ; GET THE NUMBER
1582          MOV      (SP)+,R1    ; PUT ENTRY INTO R1
1583          BNE     2$          ; BR IF NOT ZERO
1584          JMP     TST1        ; ENTER ZERO - PERFORM ALL TESTS
1585          CMP     R1,MAXTN    ; SEE IF NUMBER GREATER THAN MAXIMUM
1586          BLE     3$          ; BR IF LESS OR EQUAL
1587          TYPE      ,BADNO    ; BAD ENTRY
1588          BR      1$          ; TRY AGAIN
1589          DEC     R1          ; DECREMENT ENTRY
1590          ASL     R1          ; SHIFT IT LEFT
1591          MOV     TSTADR(R1),4$ ; GET THE TEST ADDRESS
1592          INC     KYBCTL      ; SET SINGLE TEST INDICATOR
1593          MOV     #1,$ICNT    ; PRESET ITERATION COUNT
1594          JMP     24$         ; GO TO THE SELECTED TEST
1595          .WORD   0           ; TEST ADDRESS GOES HERE
1596
1597          ;CHANGE THE RH11 UNIBUS ADDRESS USED BY THE PROGRAM
1598
1599          CHANGE: TST     CHGADR ; CHANGE THE ADDRESS ?
1600          BEQ     3$          ; BR IF NOT
1601          CLR     CHGADR      ; CLEAR THE INDICATOR
1602          TYPE   ADDRIS      ; TYPE OUT WHAT THE PRESENT ADDRESS IS
1603          MOV    $RPNADR,-(SP) ; PUT THE ADDRESS ON THE STACK
1604          TYPOC          ; TYPE THE ACTUAL ADDRESS
  
```

F03

```

1605 002622 104401 001207 TYPE ,SCLF ;CR-LF
1606 002626 104401 023660 TYPE ,NTRH11 ;ASK FOR NEW ADDRESS
1607 002632 104412 RDOCT
1608 002634 005716 TST (SP) ;0 OR 'CR' ENTERED ?
1609 002636 001402 BEQ 2$ ;BR IF EITHER ENTERED (NO ADDRESS CHANGE)
1610 002640 011637 001272 MOV (SP),SRPADR ;NEW RH11 ADDRESS
1611 002644 005726 2$: TST (SP)+ ;CORRECT THE STACK POINTER
1612 002646 012737 002666 000004 3$: MOV #4$,2#4 ;LOAD TRAP ADDRESS
1613 002654 013700 001272 MOV SRPADR,RO ;RH11 ADDRESS
1614 002660 005760 000002 TST RPOW(RO) ;SEE IF RH11 RESPONDS AT THAT ADDRESS
1615 002664 000404 BR 5$ ;BR, RH11 ALIVE AT PRESENT ADDRESS
1616 002666 104025 4$: ERROR 25 ;NO RESPONSE TO ADDRESS
1617 002670 062706 000004 ADD #4,SP ;RESET THE STACK POINTER
1618 002674 000745 BR 1$ ;GET ADDRESS AGAIN
1619 002676 012737 000006 000004 5$: MOV #6,2#4 ;RESTORE THE VECTOR
1620 002704 000207 RTS PC ;RETURN

```

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

.SBTTL \*\*\* TESTS \*\*\*

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

```

1629 002706 013700 001272 TST1AA: MOV SRPADR,RO ;;RESTORE RO AFTER END OF PASS

```

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

\*TEST 1 DRIVE ACCESS TEST

\*VERIFY THAT THE DRIVE CAN BE ACCESSED THROUGH BOTH PORTS

```

* A. SELECT DRIVE, VERIFY THAT THE DRIVE IS PRESENT, THAT THE
* DRIVE IS A DUAL PORT RPO4/5/6, THAT THE DRIVE IS ONLINE (RPOS1 HAS
* 'MOL' 'PGM' 'DPR', & 'DRY' BITS SET), AND THE THE DRIVE SERIAL
* NUMBER READ THROUGH BOTH PORTS IS THE SAME.

```

\* B. THE TEST IS REPEATED THROUGH BOTH PORTS.

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

```

1644 002712 TST1:
1645 002712 005737 001266 TST KYBCTL ;PERFORMING ONLY SINGLE TESTS ?
1646 002716 001406 BEQ 2$ ;BR IF NOT
1647 002720 100002 BPL 1$ ;BR IF JUST ENTERED TEST
1648 002722 000137 002424 JMP EXEC ;RETURN & GET NEXT TEST NUMBER
1649 002726 012737 177777 001266 1$: MOV #-1,KYBCTL ;SET SINGLE TEST INDICATOR
1650 002734 112737 000001 001102 2$: MOV #1,$STNM ;TEST NUMBER
1651 002742 012737 002764 001106 MOV #TEST1,$LPADR ;LOAD LOOP ON TEST ADDRESS
1652 002750 012737 002764 001110 MOV #TEST1,$LPERR ;LOAD LOOP ON ERROR ADDRESS
1653 002756 012737 000001 001176 MOV #1,$TIMES ;DO 1 ITERATION
1654 002764 012706 001100 TEST1: MOV #STACK,SP ;SETUP THE STACK POINTER

```

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

;;VERIFY THAT DRIVE IS PRESENT THROUGH PORTS A & B

```

1659 002770 113760 001224 000010 MOV#B PORTA,RPCS2(RO) ;SELECT PORT A
1660 002776 013737 001224 001234 MOV PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT

```

```

1661 003004 005760 000012 TST RPD51(R0) ;SEE IF DRIVE (PORT A) PRESENT
1662 003010 005037 001244 CLR CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
1663 003014 016037 000010 001126 MOV RPCS2(R0), $B0ADR ;GET CONTENTS OF RPCS2
1664 003022 012737 000010 001122 MOV #RPCS2, $B0ADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1665 003030 060037 001122 ADD R0, $B0ADR ;ADD RH11 BASE ADDRESS
1666 003034 005037 001124 CLR $GDDAT ;WHAT REGISTER SHOULD BE
1667 003040 013737 001126 001164 MOV $B0ADR, $TMP0 ;MOVE REGISTER CONTENTS TO 'TMP0'
1668 003046 042737 167777 001164 BIC #1CNED, $TMP0 ;SAVE SPECIFIED BITS
1669 003054 023737 001124 001164 CMP $GDDAT, $TMP0 ;COMPARE THE BITS
1670 003062 001414 BEQ 64$ ;BR IF OK
1671 003064 013737 001126 001174 MOV $B0ADR, $TMP4 ;COPY 'BAD DATA'
1672 003072 042737 010000 001174 BIC #NED, $TMP4 ;CLEAR THE MASKED BITS
1673 003100 053737 001174 001124 BIS $TMP4, $GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
1674 003106 104001 ERROR 1 ;TYPE MESSAGE 1
1675 003110 005137 001244 COM CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
1676 003114 000240 64$: NOP
1677 003116 005737 001244 TST CKERR ;WAS 'NED' SET ?
1678 003122 001403 BEQ .+10 ;BR IF NOT
1679 003124 012760 000040 000010 MOV #CLR, RPCS2(R0) ;ISSUE MASSBUS INIT TO CLEAR 'NED'
1680 003132 113760 001226 000010 MOVB PORTB, RPCS2(R0) ;SELECT PORT B
1681 003140 013737 001226 001234 MOV PORTB, PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1682 003146 005760 000012 TST RPD51(R0) ;SEE IF DRIVE (PORT B) PRESENT
1683 003152 005037 001244 CLR CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
1684 003156 016037 000010 001126 MOV RPCS2(R0), $B0ADR ;GET CONTENTS OF RPCS2
1685 003164 012737 000010 001122 MOV #RPCS2, $B0ADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1686 003172 060037 001122 ADD R0, $B0ADR ;ADD RH11 BASE ADDRESS
1687 003176 005037 001124 CLR $GDDAT ;WHAT REGISTER SHOULD BE
1688 003202 013737 001126 001164 MOV $B0ADR, $TMP0 ;MOVE REGISTER CONTENTS TO 'TMP0'
1689 003210 042737 167777 001164 BIC #1CNED, $TMP0 ;SAVE SPECIFIED BITS
1690 003216 023737 001124 001164 CMP $GDDAT, $TMP0 ;COMPARE THE BITS
1691 003224 001414 BEQ 66$ ;BR IF OK
1692 003226 013737 001126 001174 MOV $B0ADR, $TMP4 ;COPY 'BAD DATA'
1693 003234 042737 010000 001174 BIC #NED, $TMP4 ;CLEAR THE MASKED BITS
1694 003242 053737 001174 001124 BIS $TMP4, $GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
1695 003250 104001 ERROR 1 ;TYPE MESSAGE 1
1696 003252 005137 001244 COM CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
1697 003256 000240 66$: NOP
1698 003260 005737 001244 TST CKERR ;WAS 'NED' SET ?
1699 003264 001403 BEQ .+10 ;BR IF NOT
1700 003266 012760 000040 000010 MOV #CLR, RPCS2(R0) ;ISSUE MASSBUS INIT TO CLEAR 'NED'
1701
1702 ;*****
1703 ;CONFIRM THAT DRIVE IS AN RPO4/5/6 AND IS DUAL PORT
1704
1705 003274 113760 001224 000010 MOVB PORTA, RPCS2(R0) ;SELECT PORT A
1706 003302 013737 001224 001234 MOV PORTA, PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1707 003310 005037 001244 CLR CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
1708 003314 016037 000026 001126 MOV RPD1(R0), $B0ADR ;GET CONTENTS OF RPD1
1709 003322 012737 000026 001122 MOV #RPD1, $B0ADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1710 003330 060037 001122 ADD R0, $B0ADR ;ADD RH11 BASE ADDRESS
1711 003334 012737 024020 001124 MOV #24020, $GDDAT ;WHAT REGISTER SHOULD BE
1712 003342 013737 001126 001164 MOV $B0ADR, $TMP0 ;MOVE REGISTER CONTENTS TO 'TMP0'
1713 003350 042737 000003 001164 BIC #1C177774, $TMP0 ;SAVE SPECIFIED BITS
1714 003356 023737 001124 001164 CMP $GDDAT, $TMP0 ;COMPARE THE BITS
1715 003364 001414 BEQ 68$ ;BR IF OK
1716 003366 013737 001126 001174 MOV $B0ADR, $TMP4 ;COPY 'BAD DATA'

```

# H03

1717	003374	042737	177774	001174	BIC	#177774,\$TMP4	;CLEAR THE MASKED BITS
1718	003402	053737	001174	001124	BIS	\$TMP4,\$GDDAT	;'OR' WITH GOOD DATA FOR TYPEOUT
1719	003410	104002			ERROR	2	;TYPE MESSAGE 2
1720	003412	005137	001244		COM	CKERR	;SET THE REGISTER COMPARE ERROR INDICATOR
1721	003416	000240			NOP		
1722	003420	113760	001226	000010	MOV	PORTB,RPCS2(RO)	;SELECT PORT B
1723	003426	013737	001226	001234	MOV	PORTB,PTNBR	;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1724	003434	005037	001244		CLR	CKERR	;CLEAR THE 'CHECK ERROR' INDICATOR
1725	003440	016037	000026	001126	MOV	RPDT(RO), \$BDDAT	;GET CONTENTS OF RPDT
1726	003446	012737	000026	001122	MOV	#RPDT,\$B0ADR	;FORM REGISTER ADDRESS OF ERROR MESSAGE
1727	003454	060037	001122		ADD	RO,\$B0ADR	;ADD RH11 BASE ADDRESS
1728	003460	012737	024020	001124	MOV	#24020,\$GDDAT	;WHAT REGISTER SHOULD BE
1729	003466	013737	001126	001164	MOV	\$BDDAT,\$TMP0	;MOVE REGISTER CONTENTS TO 'STMP0'
1730	003474	042737	000003	001164	BIC	#1C177774,\$TMP0	;SAVE SPECIFIED BITS
1731	003502	023737	001124	001164	CMP	\$GDDAT,\$TMP0	;COMPARE THE BITS
1732	003510	001414			BEQ	70\$	;BR IF OK
1733	003512	013737	001126	001174	MOV	\$BDDAT,\$TMP4	;COPY 'BAD DATA'
1734	003520	042737	177774	001174	BIC	#177774,\$TMP4	;CLEAR THE MASKED BITS
1735	003526	053737	001174	001124	BIS	\$TMP4,\$GDDAT	;'OR' WITH GOOD DATA FOR TYPEOUT
1736	003534	104002			ERROR	2	;TYPE MESSAGE 2
1737	003536	005137	001244		COM	CKERR	;SET THE REGISTER COMPARE ERROR INDICATOR
1738	003542	000240			NOP		
1739							
1740							
1741							
1742							
1743	003544	113760	001224	000010	MOV	PORTA,RPCS2(RO)	;SELECT PORT A
1744	003552	013737	001224	001234	MOV	PORTA,PTNBR	;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1745	003560	005037	001244		CLR	CKERR	;CLEAR THE 'CHECK ERROR' INDICATOR
1746	003564	016037	000012	001126	MOV	RPOS1(RO), \$BDDAT	;GET CONTENTS OF RPOS1
1747	003572	012737	000012	001122	MOV	#RPOS1,\$B0ADR	;FORM REGISTER ADDRESS OF ERROR MESSAGE
1748	003600	060037	001122		ADD	RO,\$B0ADR	;ADD RH11 BASE ADDRESS
1749	003604	012737	001000	001124	MOV	#PGM,\$GDDAT	;WHAT REGISTER SHOULD BE
1750	003612	013737	001126	001164	MOV	\$BDDAT,\$TMP0	;MOVE REGISTER CONTENTS TO 'STMP0'
1751	003620	042737	176777	001164	BIC	#1CPGM,\$TMP0	;SAVE SPECIFIED BITS
1752	003626	023737	001124	001164	CMP	\$GDDAT,\$TMP0	;COMPARE THE BITS
1753	003634	001414			BEQ	72\$	;BR IF OK
1754	003636	013737	001126	001174	MOV	\$BDDAT,\$TMP4	;COPY 'BAD DATA'
1755	003644	042737	001000	001174	BIC	#PGM,\$TMP4	;CLEAR THE MASKED BITS
1756	003652	053737	001174	001124	BIS	\$TMP4,\$GDDAT	;'OR' WITH GOOD DATA FOR TYPEOUT
1757	003660	104003			ERROR	3	;TYPE MESSAGE 3
1758	003662	005137	001244		COM	CKERR	;SET THE REGISTER COMPARE ERROR INDICATOR
1759	003666	000240			NOP		
1760	003670	005037	001244		CLR	CKERR	;CLEAR THE 'CHECK ERROR' INDICATOR
1761	003674	016037	000012	001126	MOV	RPOS1(RO), \$BDDAT	;GET CONTENTS OF RPOS1
1762	003702	012737	000012	001122	MOV	#RPOS1,\$B0ADR	;FORM REGISTER ADDRESS OF ERROR MESSAGE
1763	003710	060037	001122		ADD	RO,\$B0ADR	;ADD RH11 BASE ADDRESS
1764	003714	012737	010600	001124	MOV	#MOL!DPR!DRY,\$GDDAT	;WHAT REGISTER SHOULD BE
1765	003722	013737	001126	001164	MOV	\$BDDAT,\$TMP0	;MOVE REGISTER CONTENTS TO 'STMP0'
1766	003730	042737	167177	001164	BIC	#1C10600,\$TMP0	;SAVE SPECIFIED BITS
1767	003736	023737	001124	001164	CMP	\$GDDAT,\$TMP0	;COMPARE THE BITS
1768	003744	001414			BEQ	74\$	;BR IF OK
1769	003746	013737	001126	001174	MOV	\$BDDAT,\$TMP4	;COPY 'BAD DATA'
1770	003754	042737	010600	001174	BIC	#10600,\$TMP4	;CLEAR THE MASKED BITS
1771	003762	053737	001174	001124	BIS	\$TMP4,\$GDDAT	;'OR' WITH GOOD DATA FOR TYPEOUT
1772	003770	104004			ERROR	4	;TYPE MESSAGE 4

68\$:

70\$:

\*\*\*\*\*  
;VERIFY THROUGH BOTH PORTS THAT THE DRIVE IS ON LINE AND IN NEUTRAL

72\$:

```

1773 003772 005137 001244          COM      CKERR          ;SET THE REGISTER COMPARE ERROR INDICATOR
1774 003776 000240          NOP
1775 004000 113760 001226 000010 74$:  MOVB     PORTB,RPCS2(R0) ;SELECT PORT B
1776 004006 013737 001226 001234      MOV      PORTB,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1777 004014 005037 001244          CLR      CKERR        ;CLEAR THE 'CHECK ERROR' INDICATOR
1778 004020 016037 000012 001126      MOV      RPS1(R0),SBOAT ;GET CONTENTS OF RPS1
1779 004026 012737 000012 001122      MOV      #RPS1,SBOADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1780 004034 060037 001122          ADD     R0,SBOADR    ;ADD RHI1 BASE ADDRESS
1781 004040 012737 001000 001124      MCV     #PGM,$GDOAT ;WHAT REGISTER SHOULD BE
1782 004046 013737 001126 001164      MOV     $BDOAT,$TMP0 ;MOVE REGISTER CONTENTS TO 'TMP0'
1783 004054 042737 176777 001164      BIC     #1CPGM,$TMP0 ;SAVE SPECIFIED BITS
1784 004062 023737 001124 001164      CMP     $GDOAT,$TMP0 ;COMPARE THE BITS
1785 004070 001414          BEQ     76$          ;BR IF OK
1786 004072 013737 001126 001174      MOV     $BDOAT,$TMP4 ;COPY 'BAD DATA'
1787 004100 042737 001000 001174      BIC     #PGM,$TMP4   ;CLEAR THE MASKED BITS
1788 004106 053737 001174 001124      BIS     $TMP4,$GDOAT ;'OR' WITH GOOD DATA FOR TYPEOUT
1789 004114 104003          ERROR  3            ;TYPE MESSAGE 3
1790 004116 005137 001244          COM      CKERR          ;SET THE REGISTER COMPARE ERROR INDICATOR
1791 004122 000240          NOP
1792 004124 005037 001244          CLR      CKERR        ;CLEAR THE 'CHECK ERROR' INDICATOR
1793 004130 016037 000012 001126      MOV     RPS1(R0),SBOAT ;GET CONTENTS OF RPS1
1794 004136 012737 000012 001122      MOV     #RPS1,SBOADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1795 004144 060037 001122          ADD     R0,SBOADR    ;ADD RHI1 BASE ADDRESS
1796 004150 012737 010600 001124      MOV     #MOL!DPR!DRY,$GDOAT ;WHAT REGISTER SHOULD BE
1797 004156 013737 001126 001164      MOV     $BDOAT,$TMP0 ;MOVE REGISTER CONTENTS TO 'TMP0'
1798 004164 042737 167177 001164      BIC     #1C10600,$TMP0 ;SAVE SPECIFIED BITS
1799 004172 023737 001124 001164      CMP     $GDOAT,$TMP0 ;COMPARE THE BITS
1800 004200 001414          BEQ     78$          ;BR IF OK
1801 004202 013737 001126 001174      MOV     $BDOAT,$TMP4 ;COPY 'BAD DATA'
1802 004210 042737 010600 001174      BIC     #10600,$TMP4 ;CLEAR THE MASKED BITS
1803 004216 053737 001174 001124      BIS     $TMP4,$GDOAT ;'OR' WITH GOOD DATA FOR TYPEOUT
1804 004224 104004          ERROR  4            ;TYPE MESSAGE 4
1805 004226 005137 001244          COM      CKERR          ;SET THE REGISTER COMPARE ERROR INDICATOR
1806 004232 000240          NOP
1807
1808
1809 ;*****
1810 ;VERIFY THAT DRIVE SERIAL NUMBER SEEN THROUGH BOTH PORTS IS THE SAME
1811 004234 113760 001224 000010      MOVB     PORTA,RPCS2(R0) ;SELECT PORT A
1812 004242 016037 000030 001124      MOV      RPSN(R0),$GDOAT ;STORE THE PORT A SERIAL NUMBER
1813 004250 113760 001226 000010      MOVB     PORTB,RPCS2(R0) ;SELECT PORT B
1814 004256 016037 000030 001126      MOV      RPSN(R0),SBOAT ;STORE THE PORT B SERIAL NUMBER
1815 004264 023737 001124 001126      CMP      $GDOAT,$SBOAT ;ARE THEY THE SAME ?
1816 004272 001406          BEQ     1$          ;BR IF THEY ARE
1817 004274 104005          ERROR  5            ;REPORT THE ERROR
1818 004276 032777 100000 174634      BIT      #SW15,$SWR   ;HALT ON ERROR ?
1819 004304 001001          BNE     1$          ;BR IF SET - PROGRAM HAS ALREADY HALTED
1820 004306 000000          HALT
1821 004310 000004          HALT
1822
1823
1824
1825 ;*****
1826 ;TEST 2          SET 'VV' FOR PORT A
1827 ;*
1828 ;SET VOLUME VALID

```

J03

1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884

004312  
004312 005737 001266  
004316 001406  
004320 100002  
004322 000137 002424  
004326 012737 177777 001266  
004334 112737 000002 001102  
004342 012737 004364 001106  
004350 012737 004364 001110  
004356 012737 000001 001176  
004364 012706 001100  
004370 113760 001224 000010  
004376 013737 001224 001231  
  
004404 012760 000011 000000  
004412 012760 000021 000000  
004420 012760 010000 000032  
  
004426 005037 001244  
004432 016037 000012 001126  
004440 012737 000012 001122  
004446 060037 001122  
004452 012737 011700 001124  
004460 013737 001126 001164  
004466 042737 106077 001164  
004474 023737 001124 001164  
004502 001414  
004504 013737 001126 001174  
004512 042737 071700 001174  
004520 053737 001174 001124  
004526 104010  
004530 005137 001244  
004534 000240  
  
004536 113760 001224 000010

```

;*
;* A. ISSUE A DRIVE CLEAR COMMAND THROUGH PORT A.
;*
;* B. ISSUE A READIN PRESET COMMAND THROUGH PORT A. VERIFY
;* THAT THE 'VV' BIT IS SET FOR PORT A.
;*
;* C. ISSUE A RELEASE COMMAND THROUGH PORT A. VERIFY THAT
;* THE DRIVE RETURNED TO NEUTRAL AND THAT NEITHER ATTENTION
;* BIT IS SET.
*****
TST2:
TST KYBCTL ;PERFORMING ONLY SINGLE TESTS ?
BEQ 2$ ;BR IF NOT
BPL 1$ ;BR IF JUST ENTERED TEST
JMP EXEC ;RETURN & GET NEXT TEST NUMBER
1$: MOV #-1,KYBCTL ;SET SINGLE TEST INDICATOR
2$: MOVB #2,$STNM ;TEST NUMBER
MOV #TEST2,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #TEST2,$LPERR ;LOAD LOOP ON ERROR ADDRESS
MOV #1,$TIMES ;DO 1 ITERATION
TEST2: MOV #STACK,$SP ;SETUP THE STACK POINTER
MOVB PORTA,$PCS2($R0) ;SELECT PORT A
MOV PORTA,$PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
*****
;SET VOLUME VALUE FOR PORT
MOV #11,$PCS1($R0) ;ISSUE A DRIVE CLEAR
MOV #21,$PCS1($R0) ;ISSUE A READIN PRESET
MOV #FMT22,$RPOF($R0) ;SET FMT22
*****
;VERIFY THAT THE DRIVE STATUS IS CORRECT
CLR CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
MOV $RPS1($R0),$SBOADR ;GET CONTENTS OF $RPS1
MOV #RPS1,$SBOADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
ADD $R0,$SBOADR ;ADD RH11 BASE ADDRESS
MOV #MOL!PGM!DPR!DRY!VV,$GDDAT ;WHAT REGISTER SHOULD BE
MOV $SBOADR,$STMP0 ;MOVE REGISTER CONTENTS TO '$STMP0'
BIC #1C71700,$STMP0 ;SAVE SPECIFIED BITS
CMP $GDDAT,$STMP0 ;COMPARE THE BITS
BEQ 64$ ;BR IF OK
MOV $SBOADR,$STMP4 ;COPY 'BAD DATA'
BIC #71700,$STMP4 ;CLEAR THE MASKED BITS
BIS $STMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
ERROR 10 ;TYPE MESSAGE 10
COM CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
64$: NOP
*****
;RELEASE THE DRIVE FROM PORT A
MOVB PORTA,$PCS2($R0) ;SELECT PORT A

```



# K03

MC-11-DZRJF-A, RP04/5/6 DUAL CONTRLR LOGIC TEST - PART 2  
DZRJFA.CMB 02-NOV-76 19:11 T2 SET 'VV' FOR PORT A

MACY11 27(1006) 02-NOV-76 19:12 PAGE 36

```
1895 004544 013737 001224 001234 MOV PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1896 004552 012760 000013 000000 MOV #13,RPCS1(RO) ;ISSUE RELEASE THROUGH PORT A
1897
1898 ;VERIFY THAT THE DRIVE IS IN NEUTRAL
1899
1900 004560 005037 001250 CLR RELERR ;CLEAR THE 'RELEASE ERROR' INDICATOR
1901 004564 012737 000012 001122 MOV #RPDS1,$BDDADR ;FORM THE ADDRESS OF RPDS1 FOR TYPEOUT
1902 004572 060037 001122 ADD RO,$BDDADR ;ADD THE I/O BASE ADDRESS
1903 004576 012737 011600 001124 MOV #MOL!PGM!DPR!DRY,$GDDAT ;COMPARISON CONSTANT
1904 004604 113760 001224 000010 MOVB PORTA,RPCS2(RO) ;SELECT PORT A.
1905 004612 016037 000012 001170 MOV RPDS1(RO),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
1906 004620 013737 001170 001164 MOV $TMP2,$TMP0 ;COPY IT INTO '$TMP0'
1907 004626 042737 100100 001164 BIC #ATA!VV,$TMP0 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
1908 004634 113760 001226 000010 MOVB PORTB,RPCS2(RO) ;SELECT PORT B.
1909 004642 016037 000012 001172 MOV RPDS1(RO),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
1910 004650 013737 001172 001166 MOV $TMP3,$TMP1 ;COPY IT INTO '$TMP1'
1911 004656 042737 100100 001166 BIC #ATA!VV,$TMP1 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
1912 004664 023737 001164 001166 CMP $TMP0,$TMP1 ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
1913 004672 001006 BNE 66$ ;BR IF NOT
1914 004674 005737 001164 TST $TMP0 ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
1915 004700 001037 BNE 68$ ;BR IF NOT
1916 004702 104034 ERROR 34 ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
1917 004704 000137 005070 JMP 70$ ;BYPASS THE REST OF THE CHECKS
1918 004710 013737 001170 001126 66$: MOV $TMP2,$BDDAT ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
1919 004716 013737 001226 001234 MOV PORTB,PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
1920 004724 113760 001226 000010 MOVB PORTB,RPCS2(RO) ;SELECT PORT B.
1921 004732 005737 001164 TST $TMP0 ;SEE IF STATUS EQ 0 FROM PORT A.
1922 004736 001414 BEQ 67$ ;BR IF ZERO
1923 004740 013737 001224 001234 MOV PORTA,PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
1924 004746 013737 001172 001126 MOV $TMP3,$BDDAT ;'BAD DATA' FOR ERROR TYPE OUT
1925 004754 113760 001224 000010 MOVB PORTA,RPCS2(RO) ;SELECT PORT A.
1926 004762 005737 001166 TST $TMP1 ;SEE IF STATUS EQ ZERO FROM PORT B.
1927 004766 001004 BNE 68$ ;BR IF NOT
1928 004770 012737 177777 001250 67$: MOV #-1,RELERR ;SET 'RELEASE ERROR' INDICATOR
1929 004776 104036 ERROR 36 ;TYPE ERROR MESSAGE 36
1930 005000 013737 001170 001126 68$: MOV $TMP2,$BDDAT ;LOOK FOR BIT FAILURES WHEN RPDS1 READ
1931 005006 013737 001224 001234 MOV PORTA,PTNBR ;CHANGE PORT NUMBER
1932 005014 042737 100100 001170 BIC #ATA!VV,$TMP2 ;DON'T CHECK ATTN BIT OR VV BIT
1933 005022 023737 001124 001170 CMP $GDDAT,$TMP2 ;ALL BITS OK ?
1934 005030 001401 BEQ 69$ ;BR IF OK FROM PORT A.
1935 005032 104037 ERROR 37 ;REPORT ERROR
1936 005034 013737 001172 001126 69$: MOV $TMP3,$BDDAT ;CHECK RPDS1 FOR BIT FAILURES - FROM PORT B.
1937 005042 013737 001226 001234 MOV PORTB,PTNBR ;CHANGE PORT NUMBER
1938 005050 042737 100100 001172 BIC #ATA!VV,$TMP3 ;DON'T CHECK ATTN BIT OR VV BIT
1939 005056 023737 001124 001172 CMP $GDDAT,$TMP3 ;SEE IF READ OK FROM PORT B.
1940 005064 001401 BEQ 70$ ;BR IF OK
1941 005066 104037 ERROR 37 ;REPORT THE ERRJR
1942 005070 000240 70$: NOP
1943 005072 000004 SCOPE ;LOOP ?
1944
1945 *****
1946 *TEST 3 SET 'VV' FOR PORT B
1947 *
1948 *SET VOLUME VALID
1949 *
1950 * A. ISSUE A DRIVE CLEAR COMMAND THROUGH PORT B.
```

L03

```

1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
005074 005737 001266
1951 005074 005737 001266
1952 005100 001406
1953 005102 100002
1954 005104 000137 002424
1955 005110 012737 177777 001266
1956 005116 112737 000003 001102
1957 005124 012737 005146 001106
1958 005132 012737 005146 001110
1959 005140 012737 000001 001176
1960 005146 012706 001100
1961 005152 113760 001226 000010
1962 005160 013737 001226 001234
1963
1964
1965
1966
1967 005166 012760 000011 000000
1968 005174 012760 000021 000000
1969 005202 012760 010000 000032
1970
1971
1972
1973
1974 005210 005037 001244
1975 005214 016037 000012 001126
1976 005222 012737 000012 001122
1977 005230 060037 001122
1978 005234 012737 011700 001124
1979 005242 013737 001126 001164
1980 005250 042737 106077 001164
1981 005256 023737 001124 001164
1982 005264 001414
1983 005266 013737 001126 001174
1984 005274 042737 071700 001174
1985 005302 053737 001174 001124
1986 005310 104010
1987 005312 005137 001244
1988 005316 000240
1989
1990
1991
1992
1993
1994 005320 113760 001226 000010
1995 005326 013737 001226 001234
1996 005334 012760 000013 000000

```

```

*****
TST3:
TST KYBCTL ;PERFORMING ONLY SINGLE TESTS ?
BEQ 25 ;BR IF NOT
BPL 15 ;BR IF JUST ENTERED TEST
JMP EXEC ;RETURN & GET NEXT TEST NUMBER
15: MOV #-1,KYBCTL ;SET SINGLE TEST INDICATOR
25: MOVB #3,$STNM ;TEST NUMBER
MOV #TEST3,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #TEST3,$LPERR ;LOAD LOOP ON ERROR ADDRESS
MOV #1,$TIMES ;DO 1 ITERATION
TEST3: MOV #STACK,$SP ;SETUP THE STACK POINTER
MOVB PORTB,$PC52(RO) ;SELECT PORT B
MOV PORTB,$PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
*****
;SET VOLUME VALUE FOR PORT
MOV #11,$PC51(RO) ;ISSUE A DRIVE CLEAR
MOV #21,$PC51(RO) ;ISSUE A READIN PRESET
MOV #FMT22,$RPOF(RO) ;SET FMT22
*****
;VERIFY THAT THE DRIVE STATUS IS CORRECT
CLR CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
MOV $RPO51(RO),$SBDAT ;GET CONTENTS OF RPO51
MOV #RPO51,$SBOADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
ADD RO,$SBOADR ;ADD RHI1 BASE ADDRESS
MOV #MOL!PGM!DPR!DRY!VV,$GDDAT ;WHAT REGISTER SHOULD BE
MOV $SBDAT,$STMP0 ;MOVE REGISTER CONTENTS TO 'STMP0'
BIC #1C71700,$STMP0 ;SAVE SPECIFIED BITS
CMP $GDDAT,$STMP0 ;COMPARE THE BITS
BEQ 645 ;BR IF OK
MOV $SBDAT,$STMP4 ;COPY 'BAD DATA'
BIC #71700,$STMP4 ;CLEAR THE MASKED BITS
BIS $STMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
ERROR 10 ;TYPE MESSAGE 10
COM CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
645: NOP
*****
;RELEASE THE DRIVE FROM PORT B
MOVB PORTB,$PC52(RO) ;SELECT PORT B
MOV PORTB,$PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
MOV #13,$PC51(RO) ;ISSUE RELEASE THROUGH PORT B

```

M03

```

1997
1998
1999
2000 005342 005037 001250 CLR RELERR ;CLEAR THE 'RELEASE ERROR' INDICATOR
2001 005346 012737 000012 001122 MOV #RPS1,$B0ADR ;FORM THE ADDRESS OF RPS1 FOR TYPEOUT
2002 005354 060037 001122 ADD R0,$B0ADR ;ADD THE I/O BASE ADDRESS
2003 005360 012737 011600 001124 MOV #MOL!PGM!DPR!DRY,$GDDAT ;COMPARISON CONSTANT
2004 005366 113760 001224 000010 MOV#B PORTA,RPCS2(R0) ;SELECT PORT A.
2005 005374 016037 000012 001170 MOV RPS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
2006 005402 013737 001170 001164 MOV $TMP2,$TMP0 ;COPY IT INTO '$TMP0'
2007 005410 042737 100100 001164 BIC #ATA!VV,$TMP0 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2008 005416 113760 001226 000010 MOV#B PORTB,RPCS2(R0) ;SELECT PORT B.
2009 005424 016037 000012 001172 MOV RPS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
2010 005432 013737 001172 001166 MOV $TMP3,$TMP1 ;COPY IT INTO '$TMP1'
2011 005440 042737 100100 001166 BIC #ATA!VV,$TMP1 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2012 005446 023737 001164 001166 CMP $TMP0,$TMP1 ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2013 005454 001006 BNE 66$ ;BR IF NOT
2014 005456 005737 001164 TST $TMP0 ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
2015 005462 001037 BNE 68$ ;BR IF NOT
2016 005464 104034 ERROR 34 ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2017 005466 000137 005652 JMP 70$ ;BYPASS THE REST OF THE CHECKS
2018 005472 013737 001170 001126 66$: MOV $TMP2,$BDDAT ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2019 005500 013737 001226 001234 MOV PORTB,PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2020 005506 113760 001226 000010 MOV#B PORTB,RPCS2(R0) ;SELECT PORT B.
2021 005514 005737 001164 TST $TMP0 ;SEE IF STATUS EQ 0 FROM PORT A.
2022 005520 001414 BEQ 67$ ;BR IF ZERO
2023 005522 013737 001224 001234 MOV PORTA,PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2024 005530 013737 001172 001126 MOV $TMP3,$BDDAT ;'BAD DATA' FOR ERROR TYPE OUT
2025 005536 113760 001224 000010 MOV#B PORTA,RPCS2(R0) ;SELECT PORT A.
2026 005544 005737 001166 TST $TMP1 ;SEE IF STATUS EQ ZERO FROM PORT B.
2027 005550 001004 BNE 68$ ;BR IF NOT
2028 005552 012737 177777 001250 67$: MOV #-1,RELERR ;SET 'RELEASE ERROR' INDICATOR
2029 005560 104036 ERROR 36 ;TYPE ERROR MESSAGE 36
2030 005562 013737 001170 001126 68$: MOV $TMP2,$BDDAT ;LOOK FOR BIT FAILURES WHEN RPS1 READ
2031 005570 013737 001224 001234 MOV PORTA,PTNBR ;CHANGE PORT NUMBER
2032 005576 042737 100100 001170 BIC #ATA!VV,$TMP2 ;DON'T CHECK ATTN BIT OR VV BIT
2033 005604 023737 001124 001170 CMP $GDDAT,$TMP2 ;ALL BITS OK ?
2034 005612 001401 BEQ 69$ ;BR IF OK FROM PORT A.
2035 005614 104037 ERROR 37 ;REPORT ERROR
2036 005616 013737 001172 001126 69$: MOV $TMP3,$BDDAT ;CHECK RPS1 FOR BIT FAILURES - FROM PORT B.
2037 005624 013737 001226 001234 MOV PORTB,PTNBR ;CHANGE PORT NUMBER
2038 005632 042737 100100 001172 BIC #ATA!VV,$TMP3 ;DON'T CHECK ATTN BIT OR VV BIT
2039 005640 023737 001124 001172 CMP $GDDAT,$TMP3 ;SEE IF READ OK FROM PORT B.
2040 005646 001401 BEQ 70$ ;BR IF OK
2041 005650 104037 ERROR 37 ;REPORT THE ERROR
2042 005652 000240 70$: NOP
2043 005654 000004 SCOPE ;LOOP ?
2044
2045
2046
2047
2048
2049 *****
2050 *TEST 4 MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT A
2051 *
2052 *MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT A
*
```

N03

2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108

005656  
005656 005737 001266  
005652 001406  
005664 100002  
005666 000137 002424  
005672 012737 177777 001266  
005700 112737 000004 001102  
005706 012737 005730 001106  
005714 012737 005730 001110  
005722 012737 000001 001176  
005730 012706 001100  
005734 005037 001256  
005740 005037 001260  
  
005744 005037 001252  
005750 012737 003720 001254  
  
005756 113760 001224 000010  
005764 013737 001224 001236  
005772 005060 000012  
005776 113760 001226 000010  
006004 013737 001226 001234  
006012 013737 001226 001240  
006020 016037 000012 001126  
006026 010037 001122  
006032 062737 000012 001122  
006044 005037 001124  
006044 023737 001124 001126  
006052 001403  
006054 104030  
006056 000137 006572  
006062  
006062 113760 001224 000010  
006070 013737 001224 001234  
006076 016037 000012 001126  
006104 012737 011700 001124  
006112 013737 001124 001166  
006120 005137 001156  
006124 013737 001126 001164

- ;\* A. WRITE 0'S INTO RPDS1 THROUGH PORT A AND VERIFY THAT THE DRIVE HAS BEEN SEIZED.
- ;\* B. WAIT FOR TIMEOUT TO OCCUR. MEASURE THE DURATION OF THE TIMEOUT ONE-SHOT AND SAVE THE VALUE FOR LATER USE.
- ;\* C. VERIFY THAT THE TIMEOUT OCCURRED AND THAT THE DRIVE RETURNS TO NEUTRAL

\*\*\*\*\*  
TST4:

```
TST KYBCTL ;PERFORMING ONLY SINGLE TESTS ?
BEQ 25 ;BR IF NOT
BPL 15 ;BR IF JUST ENTERED TEST
JMP EXEC ;RETURN & GET NEXT TEST NUMBER
15: MOV #-1,KYBCTL ;SET SINGLE TEST INDICATOR
25: MOV #4,$STNM ;TEST NUMBER
MOV #TEST4,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #TEST4,$LPERR ;LOAD LOOP ON ERROR ADDRESS
MOV #1,$TIMES ;DO 1 ITERATION
TEST4: MOV #STACK,$SP ;SETUP THE STACK POINTER
CLR TIMEA ;CLEAR THE TIMEOUT VALUE STORAGE LOCATION
CLR TIMEAP ;CLEAR THE + 25% TOLERANCE LOCATION
```

\*\*\*\*\*  
;START THE TIMER

```
CLR TIME ;CLEAR THE ELAPSED TIME COUNTER
MOV #2000.,WATCH ;SET WATCH TO 2000 MS
```

\*\*\*\*\*

;SEIZE THE DRIVE THROUGH PORT A

```
MOV #PORTA,$RPS2 ;SELECT PORT A
MOV #PORTA,$PTNR ;STORE SEIZING PORT'S ADDRESS
CLR #RPDS1 ;WRITE RPDS1
MOV #PORTB,$RPS2 ;SELECT PORT B
MOV #PORTB,$PTNR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
MOV #PORTB,$OPPR ;'OPPOSITE' PORT ADDRESS
MOV #RPDS1,$SBOAT ;SEE IF DRIVE SEIZED BY PORT A
MOV #R0,$SBOADR ;R0!! BASE ADDRESS
ADD #RPDS1,$SBOADR ;GENERATE BAD REGISTER ADDRESS
CLR #SBOAT ;REGISTER SHOULD BE ZERO
CMP #SBOAT,$SBOAT ;IS THE REGISTER ZERO
BEQ 645 ;BR IF IT IS
ERROR 30 ;REPORT THE ERROR
JMP 45 ;BYPASS REST OF THE SUBTEST
```

645:

```
MOV #PORTA,$RPS2 ;SELECT PORT A
MOV #PORTA,$PTNR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
MOV #RPDS1,$SBOAT ;SEE IF SEIZING PORT SEES CORRECT STATUS
MOV #MOL!PGH!DPR!DRY!VV,$SBOAT ;EXPECTED STATUS
MOV #SBOAT,$STMP1 ;USE GOOD DATA AS A MASK
COM #STMP1 ;COMPLEMENT THE EXPECTED STATUS
MOV #SBOAT,$STMP0 ;SAVE THE ACTUAL STATUS
```

804

```

2109 006132 043737 001166 001154      BIC      STMP1,STMP0      ;CLEAR UNWANTED BITS
2110 006140 023737 001124 001164      CMP      $GDDAT,STMP0    ;ARE THE EXPECTED STATUS BITS SET ?
2111 006146 001401                      BEQ      65$             ;BR IF THEY ARE
2112 006150 104031                      ERROR    31             ;REPORT THE ERROR
2113 006152 000240      65$:      NOP
2114
2115      ;*****
2116      ;WAIT FOR PORT A TO TIMEOUT
2117
2118 006154 113760 001226 000010      MOV      PORTB,RPCS2(RO) ;SELECT PORT B
2119 006162 013737 001226 001234      MOV      PORTB,PTNBR    ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2120 006170 005760 000012      1$:      TST      RPS1(RO)      ;WAIT FOR THE DRIVE TO TIMEOUT
2121 006174 001006                      BNE      2$             ;BR WHEN TIMEOUT OCCURS
2122 006176 005737 001254      TST      WATCH          ;CHECK WATCH
2123 006202 001372                      BNE      1$             ;BR IF NOT ZERO
2124 006204 104006                      ERROR    6              ;NO TIMEOUT WITHIN 2 SECONDS
2125 006206 000137 006244      JMP      3$             ;BYPASS THE REST OF THE TEST
2126 006212 013737 001252 001256      2$:      MOV      TIME,TIMEA     ;SAVE THE ELAPSED TIME FOR PORT A
2127 006220 004537 020130      JSR      R5,TOLER       ;CALCULATE THE TOLERANCE
2128 006224 001256                      .WORD    TIMEA          ;TIMEOUT VALUE FOR PORT A
2129 006226 012637 001260      MOV      (SP)+,TIMEAP   ;+25% TOLERANCE
2130
2131      ;*****
2132      ;VERIFY THAT THE TIMEOUT ONE-SHOT VALUE IS AT LEAST 500 MS
2133
2134 006232 023727 001256 000764      CMP      TIMEA,#500.    ;IS TIMEOUT VALUE AT LEAST 500 MS ?
2135 006240 103001                      BHS      3$             ;BR IF IT IS
2136 006242 104007                      ERROR    7              ;TIMEOUT LESS THAN 500 MS
2137
2138      ;*****
2139      ;VERIFY THAT THE DRIVE RETURNED TO NEUTRAL AFTER PORT A TIMED OUT
2140
2141 006244      3$:
2142
2143      ;VERIFY THAT THE DRIVE IS IN NEUTRAL
2144
2145 006244 005037 001250      CLR      RELERR         ;CLEAR THE 'RELEASE ERROR' INDICATOR
2146 006250 012737 000012 001122      MOV      #RPS1,$B0ADR   ;FORM THE ADDRESS OF RPS1 FOR TYPEOUT
2147 006256 060037 001122                      ADD      RO,$B0ADR      ;ADD THE I/O BASE ADDRESS
2148 006262 012737 011700 001124      MOV      #MOL!PGM!DPR!DRY!VV,$GDDAT ;COMPARISON CONSTANT
2149 006270 113760 001224 000010      MOV      PORTA,RPCS2(RO) ;SELECT PORT A.
2150 006276 016037 000012 001170      MOV      RPS1(RO),STMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
2151 006304 013737 001170 001164      MOV      STMP2,STMP0    ;COPY IT INTO 'STMP0'
2152 006312 042737 100100 001164      BIC      #ATA!VV,STMP0  ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2153 006320 113760 001226 000010      MOV      PORTB,RPCS2(RO) ;SELECT PORT B.
2154 006326 016037 000012 001172      MOV      RPS1(RO),STMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
2155 006334 013737 001172 001166      MOV      STMP3,STMP1   ;COPY IT INTO 'STMP1'
2156 006342 042737 100100 001166      BIC      #ATA!VV,STMP1  ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2157 006350 023737 001164 001166      CMP      STMP0,STMP1   ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2158 006356 001006                      BNE      66$           ;BR IF NOT
2159 006360 005737 001164      TST      STMP0         ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
2160 006364 001015                      BNE      62$           ;BR IF NOT
2161 006366 104034                      ERROR    34            ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2162 006370 000137 006570      JMP      70$           ;BYPASS THE REST OF THE CHECKS
2163 006374 013737 001170 001126      66$:      MOV      STMP2,$BDDAT   ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2164 006402 013737 001226 001234      MOV      PORTB,PTNBR   ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL

```

```

2165 006410 113760 001226 000010      MOVB  PORTB,RPCS2(RO) ;SELECT PORT B.
2166 005416 005737 001164              TST   $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
2167 006422 001414 67$              BEQ   67$            ;BR IF ZERO
2168 006424 013737 001224 001234      MOV   PORTA,PTNBR    ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2169 006432 013737 001172 001126      MOV   $TMP3,$BDDAT   ;'BAD DATA' FOR ERROR TYPE OUT
2170 006440 113760 001224 000010      MOVB  PORTA,RPCS2(RO) ;SELECT PORT A.
2171 006446 005737 001166              TST   $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
2172 006452 001012 68$              BNE   68$            ;BR IF NOT
2173 006454 012737 177777 001250 67$:      MOV   #-1,RELERR    ;SET 'RELEASE ERROR' INDICATOR
2174 006462 012760 000011 000000      MOV   #11,RPCS1(RO) ;CLEAR THE DRIVE
2175 006470 012760 000013 000000      MOV   #13,RPCS1(RO) ;RELEASE THE DRIVE
2176 006476 104035 68$:      ERROR 35           ;TYPE ERROR MESSAGE 35
2177 006500 013737 001170 001126 68$:      MOV   $TMP2,$BDDAT   ;LOOK FOR BIT FAILURES WHEN RPOD1 READ
2178 006506 013737 001224 001234      MOV   PORTA,PTNBR    ;CHANGE PORT NUMBER
2179 006514 042737 100000 001170      BIC   #ATA,$TMP2     ;DON'T CHECK THE ATTN BIT
2180 006522 023737 001124 001170      CMP   $GDDAT,$TMP2  ;ALL BITS OK ?
2181 006530 001401 69$:      BEQ   69$            ;BR IF OK FROM PORT A.
2182 006532 104037 69$:      ERROR 37           ;REPORT ERROR
2183 006534 013737 001172 001126 69$:      MOV   $TMP3,$BDDAT   ;CHECK RPOD1 FOR BIT FAILURES - FROM PORT B.
2184 006542 013737 001226 001234      MOV   PORTB,PTNBR    ;CHANGE PORT NUMBER
2185 006550 042737 100000 001172      BIC   #ATA,$TMP3     ;DON'T CHECK THE ATTN BIT
2186 006556 023737 001124 001172      CMP   $GDDAT,$TMP3  ;SEE IF READ OK FROM PORT B.
2187 006564 001401 70$:      BEQ   70$            ;BR IF OK
2188 006566 104037 70$:      ERROR 37           ;REPORT THE ERROR
2189 006570 000240 4$:      NOP
2190 006572 000004 4$:      SCOPE ;LOOP ?

```

\*\*\*\*\*  
\*TEST 5 MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT B  
\*\*\*\*\*

- \*MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT B
- \* A. WRITE 0'S INTO RPOD1 THROUGH PORT B AND VERIFY THAT THE DRIVE HAS BEEN SEIZED.
- \* B. WAIT FOR TIMEOUT TO OCCUR. MEASURE THE DURATION OF THE TIMEOUT ONE-SHOT AND SAVE THE VALUE FOR LATER USE.
- \* C. VERIFY THAT THE TIMEOUT OCCURRED AND THAT THE DRIVE RETURNS TO NEUTRAL

\*\*\*\*\*  
\*TESTS:  
\*\*\*\*\*

```

2208 006574 005737 001266      TST   KYBCTL         ;PERFORMING ONLY SINGLE TESTS ?
2209 006574 001406 2$:      BEQ   2$            ;BR IF NOT
2210 006600 100002 1$:      BPL   1$            ;BR IF JUST ENTERED TEST
2211 006602 000137 002424      JMP   EXEC           ;RETURN & GET NEXT TEST NUMBER
2212 006604 012737 177777 001266 1$:      MOV   #-1,KYBCTL    ;SET SINGLE TEST INDICATOR
2213 006610 012737 000005 001102 2$:      MOVB  #5,$STSTM     ;TEST NUMBER
2214 006616 012737 006646 001106      MOV   #TEST5,$LPADR ;LOAD LOOP ON TEST ADDRESS
2215 006624 012737 006646 001110      MOV   #TEST5,$LPERR ;LOAD LOOP ON ER. OR ADDRESS
2216 006632 012737 000001 001176      MOV   #1,$TIMES     ;DO 1 ITERATION
2217 006640 012737 001100      MOV   #STACK,$SP    ;SETUP THE STACK POINTER
2218 006646 005037 001262      CLR   TIMEB         ;CLEAR THE TIMEOUT VALUE STORAGE LOCATION
2219 006652 005037 001264      CLR   TIMEBP        ;CLEAR THE + 25% TOLERANCE LOCATION
2220 006656 005037 001264

```



```

2221
2222
2223      ;*****
2224      ;START THE TIMER
2225 006662 005037 001252      CLR      TIME      ;CLEAR THE ELAPSED TIME COUNTER
2226 006656 012737 003720 001254  MOV      #2000.,WATCH ;SET WATCH TO 2000 MS
2227
2228      ;*****
2229
2230      ;SEIZE THE DRIVE THROUGH PORT B
2231
2232 006671 113760 001226 000010  MOVB     PORTB,RPCS2(RO) ;SELECT PORT B
2233 006702 013737 001226 001236  MOV      PORTB,SEIZPT ;STORE SEIZING PORT'S ADDRESS
2234 006710 005060 000012      CLR      RPS1(RO)      ;WRITE RPS1
2235 006714 113760 001224 000010  MOVB     PORTA,RPCS2(RO) ;SELECT PORT A
2236 006722 013737 001224 001234  MOV      PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2237 006730 013737 001224 001240  MOV      PORTA,OPPRT ;'OPPOSITE' PORT ADDRESS
2238 006736 016037 000012 001126  MOV      RPS1(RO),SBDAT ;SEE IF DRIVE SEIZED BY PORT B
2239 006744 010037 001122      MOV      RO,SBDADR ;RHLI BASE ADDRESS
2240 006750 062737 000012 001122  ADD      #RPS1,SBDADR ;GENERATE BAD REGISTER ADDRESS
2241 006756 005037 001124      CLR      $GDDAT ;REGISTER SHOULD BE ZERO
2242 006762 023737 001124 001126  CMP      $GDDAT,SBDAT ;IS THE REGISTER ZERO
2243 006770 001403      BEQ      64$ ;BR IF IT IS
2244 006772 104030      ERROR   30 ;REPORT THE ERROR
2245 006774 100137 007510      JMP      4$ ;BYPASS REST OF THE SUBTEST
2246 007000
2247 007000 113760 001226 000010 64$: MOVB     PORTB,RPCS2(RO) ;SELECT PORT B
2248 007006 013737 001226 001234  MOV      PORTB,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2249 007014 016037 000012 001126  MOV      RPS1(RO),SBDAT ;SEE IF SEIZING PORT SEES CORRECT STATUS
2250 007022 012737 011700 001124  MOV      #MOL!PGM!DPR:DRY!VV,$GDDAT ;EXPECTED STATUS
2251 007030 013737 001124 001166  MOV      $GDDAT,$TMP1 ;USE GOOD DATA AS A MASK
2252 007036 005137 001166      COM      $TMP1 ;COMPLEMENT THE EXPECTED STATUS
2253 007042 013737 001126 001164  MOV      SBDAT,$TMP0 ;SAVE THE ACTUAL STATUS
2254 007050 043737 001166 001164  BIC      $TMP1,$TMP0 ;CLEAR UNWANTED BITS
2255 007056 021737 001124 001164  CMP      $GDDAT,$TMP0 ;ARE THE EXPECTED STATUS BITS SET ?
2256 007064 001401      BEQ      65$ ;BR IF THEY ARE
2257 007066 104031      ERROR   31 ;REPORT THE ERROR
2258 007070 000240 65$: NOP
2259
2260      ;*****
2261      ;WAIT FOR PORT B TO TIMEOUT
2262
2263 007072 113760 001224 000010  MOVB     PORTA,RPCS2(RO) ;SELECT PORT A
2264 007100 013737 001224 001234  MOV      PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2265 007106 005760 000012 1$: TST     RPS1(RO) ;WAIT FOR THE DRIVE TO TIMEOUT
2266 007112 001006      BNE     2$ ;BR WHEN TIMEOUT OCCURS
2267 007114 005737 001254  TST     WATCH ;CHECK WATCH
2268 007120 001372      BNE     1$ ;BR IF NOT ZERO
2269 007122 104003      ERROR   6 ;NO TIMEOUT WITHIN 2 SECONDS
2270 007124 000137 007162      JMP     3$ ;BYPASS THE REST OF THE TEST
2271 007130 013737 001252 001262 2$: MOV     TIME,TIMEB ;SAVE THE ELAPSED TIME FOR PORT B
2272 007136 004537 020130      JSR     RS,TOLER ;CALCULATE THE TOLERANCE
2273 007142 001262      .WORD  TIMEB ;TIMEOUT VALUE FOR PORT B
2274 007144 012637 001264      MOV     (SP)+,TIMEBP ;+25% TOLERANCE
2275
2276      ;*****

```

E04

```

2277 ;VERIFY THAT THE TIMEOUT ONE-SHOT VALUE IS AT LEAST 500 MS
2278
2279 007150 023727 001262 000764 CMP TIMEB,#500. ;IS TIMEOUT VALUE AT LEAST 500 MS ?
2280 007156 103001 BHS 35 ;BR IF IT IS
2281 007160 104007 ERROR 7 ;TIMEOUT LESS THAN 500 MS
2282
2283 ;*****
2284 ;VERIFY THAT THE DRIVE RETURNED TO NEUTRAL AFTER PORT B TIMED OUT
2285
2286 007162 35:
2287
2288 ;VERIFY THAT THE DRIVE IS IN NEUTRAL
2289
2290 007162 005037 001250 CLR RELERR ;CLEAR THE 'RELEASE ERROR' INDICATOR
2291 007166 012737 000012 001122 MOV #RPS1,$B0ADR ;FORM THE ADDRESS OF RPS1 FOR TYPEOUT
2292 007174 060037 001123 ADD R0,$B0ADR ;ADD THE I/O BASE ADDRESS
2293 007200 012737 011700 001124 MOV #M0L!PGM!DPR!DRY!VV,$G0DAT ;COMPARISON CONSTANT
2294 007206 113760 001224 000010 MOVB PORTA,RPCS2(R0) ;SELECT PORT A.
2295 007214 016037 000012 001170 MOV RPS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
2296 007222 013737 001170 001164 MOV $TMP2,$TMP0 ;COPY IT INTO '$TMP0'
2297 007230 042737 100100 001164 BIC #ATA!VV,$TMP0 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2298 007236 113760 001226 000010 MOVB PORTB,RPCS2(R0) ;SELECT PORT B.
2299 007244 016037 000012 001172 MOV RPS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
2300 007252 013737 001172 001166 MOV $TMP3,$TMP1 ;COPY IT INTO '$TMP1'
2301 007260 042737 100100 001166 BIC #ATA!VV,$TMP1 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2302 007266 023737 001164 001166 CMP $TMP0,$TMP1 ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2303 007274 001006 BNE 66$ ;BR IF NOT
2304 007276 005737 001164 TST $TMP0 ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
2305 007302 001045 BNE 68$ ;BR IF NOT
2306 007304 104034 ERROR 34 ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2307 007306 000137 007506 JMP 70$ ;BYPASS THE REST OF THE CHECKS
2308 007312 013737 001170 001126 66$: MOV $TMP2,$B0DAT ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2309 007320 013737 001226 001234 MOV PORTB,PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2310 007326 113760 001226 000010 MOVB PORTB,RPCS2(R0) ;SELECT PORT B.
2311 007334 005737 001164 TST $TMP0 ;SEE IF STATUS EQ 0 FROM PORT A.
2312 007340 001414 BEQ 67$ ;BR IF ZERO
2313 007342 013737 001224 001234 MOV PORTA,PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2314 007350 013737 001172 001126 MOV $TMP3,$G0DAT ;'BAD DATA' FOR ERROR TYPE OUT
2315 007356 113760 001224 000010 MOVB PORTA,RPCS2(R0) ;SELECT PORT A.
2316 007364 005737 001166 TST $TMP1 ;SEE IF STATUS EQ ZERO FROM PORT B.
2317 007370 001012 BNE 68$ ;BR IF NOT
2318 007372 012737 177777 001250 67$: MOV #-1,RELERR ;SET 'RELEASE ERROR' INDICATOR
2319 007400 012760 000011 000000 MOV #11,RPCS1(R0) ;CLEAR THE DRIVE
2320 007406 012760 000013 000000 MOV #13,RPCS1(R0) ;RELEASE THE DRIVE
2321 007414 104035 ERROR 35 ;TYPE ERROR MESSAGE 35
2322 007416 013737 001170 001126 68$: MOV $TMP2,$B0DAT ;LOOK FOR BIT FAILURES WHEN RPS1 READ
2323 007424 013737 001224 001234 MOV PORTA,PTNBR ;CHANGE PORT NUMBER
2324 007432 042737 100000 001170 BIC #ATA,$TMP2 ;DON'T CHECK THE ATTN BIT
2325 007440 023737 001124 001170 CMP $G0DAT,$TMP2 ;ALL BITS OK ?
2326 007446 001401 BEQ 69$ ;BR IF OK FROM PORT A.
2327 007450 104037 ERROR 37 ;REPORT ERROR
2328 007452 013737 001172 001126 69$: MOV $TMP3,$B0DAT ;CHECK RPS1 FOR BIT FAILURES - FROM PORT B.
2329 007460 013737 001226 001234 MOV PORTB,PTNBR ;CHANGE PORT NUMBER
2330 007466 042737 100000 001172 BIC #ATA,$TMP3 ;DON'T CHECK THE ATTN BIT
2331 007474 023737 001124 001172 CMP $G0DAT,$TMP3 ;SEE IF READ OK FROM PORT B.
2332 007502 001401 BEQ 70$ ;BR IF OK

```

# F04

```

2333 007504 104037
2334 007506 000240
2335 007510 000004
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368 007512
2369 007512 005737 001266
2370 007516 001406
2371 007520 100002
2372 007522 000137 002424
2373 007526 012737 177777 001266
2374 007534 112737 000006 001102
2375 007542 012737 007564 001106
2376 007550 012737 007564 001110
2377 007556 012737 000001 001176
2378 007564 012706 001100
2379
2380
2381
2382 007570 113760 001224 000010
2383 007576 005060 000012
2384 007602 012760 000011 000000
2385 007610 012760 000013 000000
2386 007616 113760 001226 000010
2387 007624 005060 000012
2388 007630 012760 000011 000000
  
```

```

ERROR 37 ;REPORT THE ERROR
70$: NOP
4$: SCOPE ;LOOP ?
  
```

```

*****
*TEST 6 TEST UNLOAD COMMAND THROUGH PORT A
*
*VERIFY THAT THE UNLOAD COMMAND FUNCTIONS PROPERLY AND THAT A PORT
*TIMEOUT WILL NOT OCCUR WHILE THE 'GO' BIT IS SET.
*
* A. ISSUE AN UNLOAD COMMAND THROUGH PORT A; VERIFY THAT THE
* DRIVE IS SEIZED.
*
* B. WAIT THE MEASURED TIMEOUT INTERVAL + 25%; VERIFY THAT THE DRIVE
* DOES NOT TIME OUT. VERIFY THAT THE 'GO' BIT IS STILL SET AND
* THAT 'DRY' AND 'PIP' ARE NOT SET.
*
* C. REQUEST THAT THE OPERATOR PRESS THE 'STANDBY' BUTTON ON THE DRIVE.
*
* D. WHEN THE DRIVE CYCLES UP, VERIFY THAT THE DRIVE IS STILL SEIZED
* BY PORT A, THAT THE 'VV' BIT FOR PORT A IS RESET, THAT THE
* ATTENTION BIT FOR PORT A IS SET, AND THAT THE ATTENTION BIT
* FOR PORT B IS NOT SET.
*
* E. WAIT FOR THE PORT TIMEOUT TO RELEASE THE DRIVE. WHEN THE TIMEOUT
* OCCURS, VERIFY THAT THE DRIVE RETURNED TO NEUTRAL, THAT THE
* ATTENTION BIT FOR PORT A IS STILL SET, AND THAT THE ATTENTION
* BIT FOR PORT B IS NOT SET.
*
* F. VERIFY THAT THE 'VV' BIT FOR PORT B IS NOT SET.
*
* G. ISSUE A PACK ACKNOWLEDGE INSTRUCTION THROUGH BOTH PORTS.
  
```

```

*****
†TST6:
TST KYBCTL ;PERFORMING ONLY SINGLE TESTS ?
BEQ 2$ ;OR IF NOT
BPL 1$ ;OR IF JUST ENTERED TEST
JMP EXEC ;RETURN & GET NEXT TEST NUMBER
1$: MOV #1, KYBCTL ;SET SINGLE TEST INDICATOR
2$: MOVB #6, $TSTNM ;TEST NUMBER
MOV #TEST6, $LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #TEST6, $LPERR ;LOAD LOOP ON ERROR ADDRESS
MOV #1, $TIMES ;DO 1 ITERATION
TEST6: MOV #STACK, SP ;SETUP THE STACK POINTER

;CLEAR ATTENTION BITS FOR BOTH PORTS
MOVB PORTA, RPCS2(R0) ;SELECT PORT #A
CLR RPS1(R0) ;SEIZE THE DRIVE
MOV #11, RPS1(R0) ;ISSUE DRIVE CLEAR
MOV #13, RPS1(R0) ;RELEASE THE DRIVE
MOVB PORTB, RPCS2(R0) ;SELECT PORT #B
CLR RPS1(R0) ;SEIZE THE DRIVE THROUGH PORT 'B'
MOV #11, RPS1(R0) ;ISSUE DRIVE CLEAR
  
```

G04

```

2389 007636 012760 000013 000000      MOV      #13,RPCS1(RO) ;RELEASE THE DRIVE
2390 007644 113760 001224 000010      MOV      PORTA,RPCS2(RO) ;SELECT PORT A
2391 007652 113737 001224 001234      MOV      PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2392 007660 013737 001224 001236      MOV      PORTA,SEIZPT ;SEIZING PORT'S ADDRESS
2393
2394
2395 ;:*****
2396 ;DO AN UNLOAD COMMAND THROUGH PORT A
2397 007666 012760 000003 000000      MOV      #3,RPCS1(RO) ;ISSUE AN UNLOAD COMMAND THROUGH PORT A
2398 007674 013737 001260 001254      MOV      TIMEAP,WATCH ;TIMEOUT ONSHOT VALUE + 25%
2399 007702 005737 001254      1S: TST     WATCH ;FINISHED TIMEOUT ?
2400 007706 001375      BNE     1S ;BR IF NOT
2401
2402 ;:*****
2403 ;IS THE STATUS OK ?
2404
2405 007710 005037 001244      CLR      CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
2406 007714 016037 000000 001126      MOV      RPCS1(RO),SBDAT ;GET CONTENTS OF RPCS1
2407 007722 012737 000000 001122      MOV      #RPCS1,SBDADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
2408 007730 060037 001122      ADD     RO,SBDADR ;ADD RH11 BASE ADDRESS
2409 007734 012737 004001 001124      MOV      #DVA!GO,$GDOAT ;WHAT REGISTER SHOULD BE
2410 007742 013737 001126 001164      MOV      SBDAT,$TMP0 ;MOVE REGISTER CONTENTS TO 'TMP0'
2411 007750 042737 173776 001164      BIC     #1C4001,$TMP0 ;SAVE SPECIFIED BITS
2412 007756 023737 001124 001164      CMP     $GDOAT,$TMP0 ;COMPARE THE BITS
2413 007764 001414      BEQ     64S ;BR IF OK
2414 007766 013737 001126 001174      MOV      SBDAT,$TMP4 ;COPY 'BAD DATA'
2415 007774 042737 004001 001174      BIC     #4001,$TMP4 ;CLEAR THE MASKED BITS
2416 010002 053737 001174 001124      BIS     $TMP4,$GDOAT ;'OR' WITH GOOD DATA FOR TYPEOUT
2417 010010 104011      ERROR  11 ;TYPE MESSAGE 11
2418 010012 005137 001244      COM     CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
2419 010016 000240      64S: NOP
2420 010020 005037 001244      CLR     CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
2421 010024 016037 000012 001126      MOV     RPS1(RO),SBDAT ;GET CONTENTS OF RPS1
2422 010032 012737 000012 001122      MOV     #RPS1,SBDADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
2423 010040 060037 001122      ADD     RO,SBDADR ;ADD RH11 BASE ADDRESS
2424 010044 012737 004000 001124      MOV     #PIP!PGM!DPR,$GDOAT ;WHAT REGISTER SHOULD BE
2425 010052 013737 001126 001164      MOV     SBDAT,$TMP0 ;MOVE REGISTER CONTENTS TO 'TMP0'
2426 010060 042737 000177 001164      BIC     #1C177600,$TMP0 ;SAVE SPECIFIED BITS
2427 010066 023737 001124 001164      CMP     $GDOAT,$TMP0 ;COMPARE THE BITS
2428 010074 001414      BEQ     66S ;BR IF OK
2429 010076 013737 001126 001174      MOV     SBDAT,$TMP4 ;COPY 'BAD DATA'
2430 010104 042737 177600 001174      BIC     #177600,$TMP4 ;CLEAR THE MASKED BITS
2431 010112 053737 001174 001124      BIS     $TMP4,$GDOAT ;'OR' WITH GOOD DATA FOR TYPEOUT
2432 010120 104012      ERROR  12 ;TYPE MESSAGE 12
2433 010122 005137 001244      COM     CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
2434 010126 000240      66S: NOP
2435 010130 104401 023712      TYPE   STANDBY ;TYPE THE STANDBY MESSAGE
2436 010134 013746 001224      MOV     PORTA,-(SP) ;SAVE PORTA FOR TYPEOUT
2437
2438
2439 010140 104403      TYPOS  ;TYPE THE PORT NUMBER
2440 010142 001 ;GO TYPE--OCTAL ASCII
2441 010143 000 ;TYPE 1 DIGIT(S)
2442 010144 104401 001207      TYPE  ,SCLF ;SUPPRESS LEADING ZEROS
2443 ;CR-LF
2444
2445 ;:*****
2446 ;WAIT FOR 'MOL' TO SET

```

# H04

```
2445
2446 010150 032760 010000 000012 29: BIT #MOL,RPDS1(RO) ;WAIT FOR MOL TO SET
2447 010156 001774 BEQ 25 ;LOOP UNTIL 'MOL' SETS
2448 010160 012737 003720 001254 MOV #2000, WATCH ;SETUP A 2 SECOND STALL
2449 010166 113760 001226 0000:0 MOVB PORTB,RPCS2(RO) ;SELECT PORT B
2450 010174 013737 001226 001234 MOV PORTB,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2451 010202 005760 000012 39: TST RPDS1(RO) ;DRIVE TIMEOUT ?
2452 010206 001004 BNE 45 ;BR IF IT HAS
2453 010210 005737 001254 TST WATCH ;2 SECONDS ELAPSED ?
2454 010214 001372 BNE 35 ;BR IF NOT
2455 010216 104006 ERROR 6 ;NO TIMEOUT AFTER 2 SECONDS
2456
2457
2458 010220 ;*****
2459 49: ;VERIFY THAT THE DRIVE IS IN NEUTRAL
2460
2461
2462 010220 005037 001250 CLR RELERR ;CLEAR THE 'RELEASE ERROR' INDICATOR
2463 010224 012737 000012 001122 MOV #RPDS1,$BODAT ;FORM THE ADDRESS OF RPDS1 FOR TYPEOUT
2464 010232 060037 001122 ADD RO,$BODAT ;ADD THE I/O BASE ADDRESS
2465 010236 012737 011600 001124 MOV #MOL!PGM!DPR!DRY,$GDDAT ;COMPARISON CONSTANT
2466 010244 113760 001224 000010 MOVB PORTA,RPCS2(RO) ;SELECT PORT A.
2467 010252 016037 000012 001170 MOV RPDS1(RO),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
2468 010260 013737 001170 001164 MOV $TMP2,$TMP0 ;COPY IT INTO 'TMP0'
2469 010266 042737 100100 001164 BIC #ATA!VV,$TMP0 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2470 010274 113760 001226 000010 MOVB PORTB,RPCS2(RO) ;SELECT PORT B.
2471 010302 016037 000012 001172 MOV RPDS1(RO),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
2472 010310 013737 001172 001166 MOV $TMP3,$TMP1 ;COPY IT INTO 'TMP1'
2473 010316 042737 100100 001166 BIC #ATA!VV,$TMP1 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2474 010324 023737 001164 001166 CMP $TMP0,$TMP1 ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2475 010332 001006 BNE 685 ;BR IF NOT
2476 010334 005737 001164 TST $TMP0 ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
2477 010340 001037 BNE 705 ;BR IF NOT
2478 010342 104034 ERROR 34 ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2479 010344 000137 010530 JMP 725 ;BYPASS THE REST OF THE CHECKS
2480 010350 013737 001170 001126 685: MOV $TMP2,$BODAT ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2481 010356 013737 001226 001234 MOV PORTB,PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2482 010364 113760 001226 000010 MOVB PORTB,RPCS2(RO) ;SELECT PORT B.
2483 010372 005737 001164 TST $TMP0 ;SEE IF STATUS EQ 0 FROM PORT A.
2484 010376 001414 BEQ 695 ;BR IF ZERO
2485 010400 013737 001224 001234 MOV PORTA,PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2486 010406 013737 001172 001126 MOV $TMP3,$BODAT ;'BAD DATA' FOR ERROR TYPE OUT
2487 010414 113760 001224 000010 MOVB PORTA,RPCS2(RO) ;SELECT PORT A.
2488 010422 005737 001166 TST $TMP1 ;SEE IF STATUS EQ ZERO FROM PORT B.
2489 010426 001004 BNE 705 ;BR IF NOT
2490 010430 012737 177777 001250 695: MOV #1,RELERR ;SET 'RELEASE ERROR' INDICATOR
2491 010436 104013 ERROR 13 ;TYPE ERPOR MESSAGE 13
2492 010440 013737 001170 001126 705: MOV $TMP2,$BODAT ;LOOK FOR BIT FAILURES WHEN RPDS1 READ
2493 010444 013737 001224 001234 MOV PORTA,PTNBR ;CHANGE PORT NUMBER
2494 010448 042737 100100 001170 BIC #ATA!VV,$TMP2 ;DON'T CHECK ATTN BIT OR VV BIT
2495 010462 023737 001124 001170 CMP $GDDAT,$TMP2 ;ALL BITS OK ?
2496 010470 001401 BEQ 715 ;BR IF OK FROM PORT A.
2497 010472 104037 ERROR 37 ;REPORT ERROR
2498 010474 013737 001172 001126 715: MOV $TMP3,$BODAT ;CHECK RPDS1 FOR BIT FAILURES - FROM PORT B.
2499 010502 013737 001226 001234 MOV PORTB,PTNBR ;CHANGE PORT NUMBER
2500 010510 042737 100100 001172 BIC #ATA!VV,$TMP3 ;DON'T CHECK ATTN BIT OR VV BIT
```





2557  
2558  
2559  
2560  
2561  
2562  
2563  
2564  
2565  
2566  
2567  
2568  
2569  
2570  
2571  
2572  
2573  
2574  
2575  
2576  
2577  
2578  
2579  
2580  
2581  
2582  
2583  
2584  
2585  
2586  
2587  
2588  
2589  
2590  
2591  
2592  
2593  
2594  
2595  
2596  
2597  
2598  
2599  
2600  
2601  
2602  
2603  
2604  
2605  
2606  
2607  
2608  
2609  
2610  
2611  
2612

```

*****
*TEST 7          TEST UNLOAD COMMAND THROUGH PORT B
*
*VERIFY THAT THE UNLOAD COMMAND FUNCTIONS PROPERLY AND THAT A PORT
*TIMEOUT WILL NOT OCCUR WHILE THE 'GO' BIT IS SET.
*
*  A.  ISSUE AN UNLOAD COMMAND THROUGH PORT B; VERIFY THAT THE
*      DRIVE IS SEIZED.
*
*  B.  WAIT THE MEASURED TIMEOUT INTERVAL + 25%; VERIFY THAT THE DRIVE
*      DOES NOT TIME OUT. VERIFY THAT THE 'GO' BIT IS STILL SET AND
*      THAT 'DRY' AND 'PIP' ARE NOT SET.
*
*  C.  REQUEST THAT THE OPERATOR PRESS THE 'STANDBY' BUTTON ON THE DRIVE.
*
*  D.  WHEN THE DRIVE CYCLES UP, VERIFY THAT THE DRIVE IS STILL SEIZED
*      BY PORT B, THAT THE 'VV' BIT FOR PORT B IS RESET, THAT THE
*      ATTENTION BIT FOR PORT B IS SET, AND THAT THE ATTENTION BIT
*      FOR PORT A IS NOT SET.
*
*  E.  WAIT FOR THE PORT TIMEOUT TO RELEASE THE DRIVE. WHEN THE TIMEOUT
*      OCCURS, VERIFY THAT THE DRIVE RETURNED TO NEUTRAL, THAT THE
*      ATTENTION BIT FOR PORT B IS STILL SET, AND THAT THE ATTENTION
*      BIT FOR PORT A IS NOT SET.
*
*  F.  VERIFY THAT THE 'VV' BIT FOR PORT A IS NOT SET.
*
*  G.  ISSUE A PACK ACKNOWLEDGE INSTRUCTION THROUGH BOTH PORTS.
*
*****

```

```

011070
011070 005737 001266
011074 001406
011076 100002
011100 000137 002424
011104 012737 177777 001266
011112 112737 000007 001102
011120 012737 011142 001106
011126 012737 011142 001110
011134 012737 000001 001176
011142 012706 001100

```

```

*****
*TEST7:
TST      KYBCTL      ;PERFORMING ONLY SINGLE TESTS ?
BEQ      25          ;BR IF NOT
BPL      15          ;BR IF JUST ENTERED TEST
JMP      EXEC        ;RETURN & GET NEXT TEST NUMBER
15:      MOV      #-1,KYBCTL ;SET SINGLE TEST INDICATOR
25:      MOV      #7,$TSTNM ;TEST NUMBER
        MOV      #TEST7,$LPADR ;LOAD LOOP ON TEST ADDRESS
        MOV      #TEST7,$LPERR ;LOAD LOOP ON ERROR ADDRESS
        MOV      #1,$TIMES ;DO 1 ITERATION
TEST7:   MOV      #STACK,SP ;SETUP THE STACK POINTER

;CLEAR ATTENTION BITS FOR BOTH PORTS
MOV      PORTA,RPDS1(RO) ;SELECT PORT #A
CLR      RPDS1(RO)      ;SEIZE THE DRIVE
MOV      #11,RPDS1(RO)  ;ISSUE DRIVE CLEAR
MOV      #13,RPDS1(RO)  ;RELEASE THE DRIVE
MOV      PORTB,RPDS2(RO) ;SELECT PORT #B
CLR      RPDS2(RO)      ;SEIZE THE DRIVE THROUGH PORT 'B'
MOV      #11,RPDS1(RO)  ;ISSUE DRIVE CLEAR
MOV      #13,RPDS1(RO)  ;RELEASE THE DRIVE
MOV      PORTB,RPDS2(RO) ;SELECT PORT B
MOV      PORTB,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
MOV      PORTB,SEIZPT ;SEIZING PORT'S ADDRESS

```

# K04

```

2613
2614
2615      ;:*****
2616      ;DO AN UNLOAD COMMAND THROUGH PORT B
2617 011244 012760 000003 000000      MOV      #3,RPCS1(RO)      ;ISSUE AN UNLOAD COMMAND THROUGH PORT B
2618 011252 013737 001264 001254      MOV      TIMEBP,WATCH      ;TIMEOUT ONESHOT VALUE + 25%
2619 011260 005737 001254      15:     TST      WATCH      ;FINISHED TIMEOUT ?
2620 011264 001375      BNE      15      ;BR IF NOT
2621
2622      ;:*****
2623      ;IS THE STATUS OK ?
2624
2625 011266 005037 001244      CLR      CKERR      ;CLEAR THE 'CHECK ERROR' INDICATOR
2626 011272 016037 000000 001126      MOV      RPCS1(RO),SBDAT ;GET CONTENTS OF RPCS1
2627 011300 012737 000000 001122      MOV      #RPCS1,SBDADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
2628 011306 060037 001122      ADD      RO,SBDADR      ;ADD RH11 BASE ADDRESS
2629 011312 012737 004001 001124      MOV      #DVA!GO,$GDDAT ;WHAT REGISTER SHOULD BE
2630 011320 013737 001126 001164      MOV      SBDAT,$TMP0      ;MOVE REGISTER CONTENTS TO '$TMP0'
2631 011326 042737 173776 001164      BIC      #1C4001,$TMP0 ;SAVE SPECIFIED BITS
2632 011334 023737 001124 001164      CMP      $GDDAT,$TMP0 ;COMPARE THE BITS
2633 011342 001414      BEQ      64$      ;BR IF OK
2634 011344 013737 001126 001174      MOV      SBDAT,$TMP4 ;COPY 'BAD DATA'
2635 011352 042737 004001 001174      BIC      #4001,$TMP4 ;CLEAR THE MASKED BITS
2636 011360 053737 001174 001124      BIS      $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
2637 011366 104011      ERROR   11      ;TYPE MESSAGE 11
2638 011370 005137 001244      COM      CKERR      ;SET THE REGISTER COMPARE ERROR INDICATOR
2639 011374 000240      64$:     NOP
2640 011376 005037 001244      CLR      CKERR      ;CLEAR THE 'CHECK ERROR' INDICATOR
2641 011402 016037 000012 001126      MOV      RPDS1(RO),SBDAT ;GET CONTENTS OF RPDS1
2642 011410 012737 000012 001122      MOV      #RPDS1,SBDADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
2643 011416 060037 001122      ADD      RO,SBDADR      ;ADD RH11 BASE ADDRESS
2644 011422 012737 021400 001124      MOV      #PIP!PGM!DPR,$GDDAT ;WHAT REGISTER SHOULD BE
2645 011430 013737 001126 001164      MOV      SBDAT,$TMP0 ;MOVE REGISTER CONTENTS TO '$TMP0'
2646 011436 042737 000177 001164      BIC      #1C177600,$TMP0 ;SAVE SPECIFIED BITS
2647 011444 023737 001124 001164      CMP      $GDDAT,$TMP0 ;COMPARE THE BITS
2648 011452 001414      BEQ      66$      ;BR IF OK
2649 011454 013737 001126 001174      MOV      SBDAT,$TMP4 ;COPY 'BAD DATA'
2650 011462 042737 177600 001174      BIC      #177600,$TMP4 ;CLEAR THE MASKED BITS
2651 011470 053737 001174 001124      BIS      $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
2652 011476 104012      ERROR   12      ;TYPE MESSAGE 12
2653 011500 005137 001244      COM      CKERR      ;SET THE REGISTER COMPARE ERROR INDICATOR
2654 011504 000240      66$:     NOP
2655 011506 104401 023712      TYPE     STANDBY      ;TYPE THE STANDBY MESSAGE
2656 011512 013746 001226      MOV      PORTB,-(SP) ;SAVE PORTB FOR TYPEOUT
2657
2658      TYPOS
2659      .BYTE 1      ;TYPE THE PORT NUMBER
2660      .BYTE 0      ;GO TYPE--OCTAL ASCII
2661      .BYTE 0      ;TYPE 1 DIGIT(S)
2662      TYPE ,SCLF ;SUPPRESS LEADING ZEROS
2663
2664      ;:*****
2665      ;WAIT FOR 'MOL' TO SET
2666 011526 032760 010000 000012 25:     BIT      #MOL,RPDS1(RO) ;WAIT FOR MOL TO SET
2667 011534 001774      BEQ      25      ;LOOP UNTIL 'MOL' SETS
2668 011536 012737 003720 001254      MOV      #2000.,WATCH ;SETUP A 2 SECOND STALL

```

L04

```

2669 011544 113760 001224 000010      MOV  PORTA,RPCS2(RO) ;SELECT PORT A
2670 011552 013737 001224 001234      MOV  PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2671 011560 005760 000012      3$:  TST  RPDS1(RO) ;DRIVE TIMEDOUT ?
2672 011564 001004      BNE  4$ ;BR IF IT HAS
2673 011566 005737 001254      TST  WATCH ;2 SECONDS ELAPSED ?
2674 011572 001372      BNE  3$ ;BR IF NOT
2675 011574 104006      ERROR 6 ;NO TIMEOUT AFTER 2 SECONDS
2676
2677
2678 011576      ;*****
2679      4$:
2680      ;VERIFY THAT THE DRIVE IS IN NEUTRAL
2681
2682 011576 005037 001250      CLR  RELERR ;CLEAR THE 'RELEASE ERROR' INDICATOR
2683 011602 012737 000012 001122      MOV  #RPDS1,$BDDAT ;FORM THE ADDRESS OF RPDS1 FOR TYPEOUT
2684 011610 060037 001122      ADD  RO,$BDDAT ;ADD THE I/O BASE ADDRESS
2685 011614 012737 011600 001124      MOV  #MOL!PGM!DPR!DRY,$GDDAT ;COMPARISON CONSTANT
2686 011622 113760 001224 000010      MOV  PORTA,RPCS2(RO) ;SELECT PORT A.
2687 011630 016037 000012 001170      MOV  RPDS1(RO),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
2688 011636 013737 001170 001164      MOV  $TMP2,$TMP0 ;COPY IT INTO '$TMP0'
2689 011644 042737 100100 001164      BIC  #ATA!VV,$TMP0 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2690 011652 113760 001226 000010      MOV  PORTB,RPCS2(RO) ;SELECT PORT B.
2691 011660 016037 000012 001172      MOV  RPDS1(RO),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
2692 011666 013737 001172 001166      MOV  $TMP3,$TMP1 ;COPY IT INTO '$TMP1'
2693 011674 042737 100100 001166      BIC  #ATA!VV,$TMP1 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2694 011702 023737 001164 001166      CMP  $TMP0,$TMP1 ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2695 011710 001006      BNE  68$ ;BR IF NOT
2696 011712 005737 001164      TST  $TMP0 ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
2697 011716 001037      BNE  70$ ;BR IF NOT
2698 011720 104034      ERROR 34 ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2699 011722 000137 012106      JMP  72$ ;BYPASS THE REST OF THE CHECKS
2700 011726 013737 001170 001126 68$:  MOV  $TMP2,$BDDAT ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2701 011734 013737 001226 001234      MOV  PORTB,PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2702 011742 113760 001226 000010      MOV  PORTB,RPCS2(RO) ;SELECT PORT B.
2703 011750 005737 001164      TST  $TMP0 ;SEE IF STATUS EQ 0 FROM PORT A.
2704 011754 001414      BEQ  69$ ;BR IF ZERO
2705 011756 013737 001224 001234      MOV  PORTA,PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2706 011764 013737 001172 001126      MOV  $TMP3,$BDDAT ;'BAD DATA' FOR ERROR TYPE CUT
2707 011772 113760 001224 000010      MOV  PORTA,RPCS2(RO) ;SELECT PORT A.
2708 012000 005737 001166      TST  $TMP1 ;SEE IF STATUS EQ ZERO FROM PORT B.
2709 012004 001004      BNE  70$ ;BR IF NOT
2710 012006 012737 177777 001250 69$:  MOV  #-1,RELERR ;SET 'RELEASE ERROR' INDICATOR
2711 012014 104013      ERROR 13 ;TYPE ERROR MESSAGE 13
2712 012016 013737 001170 001126 70$:  MOV  $TMP2,$BDDAT ;LOOK FOR BIT FAILURES WHEN RPDS1 READ
2713 012024 013737 001224 001234      MOV  PORTA,PTNBR ;CHANGE PORT NUMBER
2714 012032 042737 100100 001170      BIC  #ATA!VV,$TMP2 ;DON'T CHECK ATTN BIT OR VV BIT
2715 012040 023737 001124 001170      CMP  $GDDAT,$TMP2 ;ALL BITS OK ?
2716 012046 001401      BEQ  71$ ;BR IF OK FROM PORT A.
2717 012050 104037      ERROR 37 ;REPORT ERROR
2718 012052 013737 001172 001126 71$:  MOV  $TMP3,$BDDAT ;CHECK RPDS1 FOR BIT FAILURES - FROM PORT B.
2719 012060 013737 001226 001234      MOV  PORTB,PTNBR ;CHANGE PORT NUMBER
2720 012066 042737 100100 001172      BIC  #ATA!VV,$TMP3 ;DON'T CHECK ATTN BIT OR VV BIT
2721 012074 023737 001124 001172      CMP  $GDDAT,$TMP3 ;SEE IF READ OK FROM PORT B.
2722 012102 001401      BEQ  72$ ;BR IF OK
2723 012104 104037      ERROR 37 ;REPORT THE ERROR
2724 012106 000240      72$:  NOP

```

M04

2725  
2726  
2727  
2728 012110 113760 001224 000010  
2729 012116 013737 001224 001234  
2730 012124 005037 001244  
2731 012130 016037 000012 001126  
2732 012136 012737 000012 001122  
2733 012144 060037 001122  
2734 012150 005037 001124  
2735 012154 013737 001126 001164  
2736 012162 042737 077777 001164  
2737 012170 023737 001124 001164  
2738 01217E 001414  
2739 012200 013737 001126 001174  
2740 012206 042737 100000 001174  
2741 012214 053737 001174 001124  
2742 012222 104014  
2743 012224 005137 001244  
2744 012230 000240  
2745 012232 113760 001226 000010  
2746 012240 013737 001226 001234  
2747 012246 005037 001244  
2748 012252 016037 000012 001126  
2749 012260 012737 000012 001122  
2750 012266 060037 001122  
2751 012272 012737 100000 001124  
2752 012300 013737 001126 001164  
2753 012306 042737 077777 001164  
2754 012314 023737 001124 001164  
2755 012322 001414  
2756 012324 013737 001126 001174  
2757 012332 042737 100000 001174  
2758 012340 053737 001174 001124  
2759 012346 104015  
2760 012350 005137 001244  
2761 012354 000240  
2762  
2763  
2764  
2765  
2766 012356 012760 000011 000000  
2767 012364 012760 000021 000000  
2768 012372 012760 000013 000000  
2769 012400 113760 001224 000010  
2770 012406 012760 000021 000000  
2771 012414 012760 010000 000032  
2772 012422 012760 000013 000000  
2773 012430 012737 011E10 001254  
2774 012436 005737 001254  
2775 012442 001375  
2776 012444 000004  
2777  
2778  
2779  
2780

```

;*****
;CHECK THE ATTENTION BITS
MOV      PORTA,RPCS2(RO) ;SELECT PORT A
MOV      PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
CLR      CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
MOV      RPS1(RO),SBDAT ;GET CONTENTS OF RPS1
MOV      #RPS1,SBDADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
ADD      RO,SBDADR ;ADD RH11 BASE ADDRESS
CLR      $GDDAT ;WHAT REGISTER SHOULD BE
MOV      SBDAT,$TMP0 ;MOVE REGISTER CONTENTS TO '$TMP0'
BIC      #1CATA,$TMP0 ;SAVE SPECIFIED BITS
CMP      $GDDAT,$TMP0 ;COMPARE THE BITS
BEQ      73$ ;BR IF OK
MOV      SBDAT,$TMP4 ;COPY 'BAD DATA'
BIC      #ATA,$TMP4 ;CLEAR THE MASKED BITS
BIS      $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
ERROR    14 ;TYPE MESSAGE 14
COM      CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
73$:    NOP
MOV      PORTB,RPCS2(RO) ;SELECT PORT B
MOV      PORTB,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
CLR      CKERR ;CLEAR THE 'CHECK ER: 3' INDICATOR
MOV      RPS1(RO),SBDAT ;GET CONTENTS OF RPS1
MOV      #RPS1,SBDADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
ADD      RO,SBDADR ;ADD RH11 BASE ADDRESS
MOV      #ATA,$GDDAT ;WHAT REGISTER SHOULD BE
MOV      SBDAT,$TMP0 ;MOVE REGISTER CONTENTS TO '$TMP0'
BIC      #1CATA,$TMP0 ;SAVE SPECIFIED BITS
CMP      $GDDAT,$TMP0 ;COMPARE THE BITS
BEQ      75$ ;BR IF OK
MOV      SBDAT,$TMP4 ;COPY 'BAD DATA'
BIC      #ATA,$TMP4 ;CLEAR THE MASKED BITS
BIS      $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
ERROR    15 ;TYPE MESSAGE 15
COM      CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
75$:    NOP
;*****
;SET 'VV' FOR EACH PORT
MOV      #11,RPCS1(RO) ;CLEAR THE DRIVE
MOV      #21,RPCS1(RO) ;DO A READIN PRESET THROUGH PORT B
MOV      #13,RPCS1(RO) ;RELEASE THE DRIVE
MOV      PORTA,RPCS2(RO) ;SELECT PORT A
MOV      #21,RPCS1(RO) ;DO A READIN PRESET THROUGH PORT A
MOV      #FMT22,RPOF(RO) ;SET 'FMT22'
MOV      #13,RPCS1(RO) ;RELEASE THE DRIVE
MOV      #5000.,WATCH ;MOTOR 'COOL DOWN' DELAY
SS:     TST      WATCH ;FINISHED ?
        BNE     SS ;BR IF NOT
        SCOPE ;LOOP ?
;*****

```

2781  
2782  
2783  
2784  
2785  
2786  
2787  
2788  
2789  
2790  
2791  
2792  
2793  
2794  
2795  
2796  
2797  
2798  
2799  
2800  
2801  
2802  
2803  
2804  
2805  
2806  
2807  
2808  
2809  
2810  
2811  
2812  
2813  
2814  
2815  
2816  
2817  
2818  
2819  
2820  
2821  
2822  
2823  
2824  
2825  
2826  
2827  
2828  
2829  
2830  
2831  
2832  
2833  
2834  
2835  
2836

\*TEST 10 TEST 'CONTROLLER SELECT' SWITCH THROUGH PORT A DURING UNLOAD  
\*  
\*VERIFY THAT THE 'CONTROLLER SELECT' SWITCH IS INHIBITED WHEN THE  
\* RP04 IS CYCLED DOWN BY AN UNLOAD COMMAND.  
\*  
\* A. ISSUE AN UNLOAD COMMAND THROUGH PORT A.  
\*  
\* B. REQUEST THAT THE OPERATOR SWITCH THE 'CONTROLLER SELECT' SWITCH  
\* TO A AND THEN PRESS THE 'STANDBY' BUTTON.  
\*  
\* C. WAIT FOR 'MOL' TO SET BY MONITORING RPDS1 THROUGH PORT  
\* A. WHEN THE DRIVE HAS CYCLED UP ('MOL' HAS SET), ISSUE  
\* RELEASE COMMAND THROUGH PORT A.  
\*  
\* D. VERIFY THAT THE DRIVE CAN BE ACCESSED BY BOTH PORTS AND THAT  
\* THE DRIVE IS STILL IN NEUTRAL.  
\*  
\* E. REQUEST THAT THE OPERATOR RETURN THE 'CONTROLLER SELECT' SWITCH  
\* TO A/B'.  
\*  
\*\*\*\*\*

012446  
012446 005737 001266  
012452 001406  
012454 100002  
012456 002137 002424  
012462 012737 177777 001266  
012470 112737 000010 001102  
012476 012737 012520 001106  
012504 012737 012520 001110  
012512 012737 000001 001176  
012520 012706 001100  
  
012524 113760 001224 000010  
012532 005060 000012  
012536 012760 000011 000000  
012544 012760 000013 000000  
012552 113760 001226 000010  
012560 005060 000012  
012564 012760 000011 000000  
012572 012760 000013 000000  
012600 113760 001224 000010  
012606 013737 001224 001234  
012614 013737 001224 001236  
012622 012760 000003 000000  
012630 104401 024124  
012634 104401 023712  
012640 013746 001224  
  
012644 104403  
012646 001  
012647 000  
012650 104401 001207  
012654 032760 010000 000012

TST10: TST KYBCTL ;PERFORMING ONLY SINGLE TESTS ?  
BEQ 25 ;BR IF NOT  
BPL 15 ;BR IF JUST ENTERED TEST  
JMP EXEC ;RETURN & GET NEXT TEST NUMBER  
15: MOV #-1,KYBCTL ;SET SINGLE TEST INDICATOR  
25: MOVB #10,\$STNM ;TEST NUMBER  
MOV #TEST10,\$LPADR ;LOAD LOOP ON TEST ADDRESS  
MOV #TEST10,\$LPERR ;LOAD LOOP ON ERROR ADDRESS  
MOV #1,\$TIMES ;DO 1 ITERATION  
TEST10: MOV #STACK,\$SP ;SETUP THE STACK POINTER  
  
;CLEAR ATTENTION BITS FOR BOTH PORTS  
MOVB PORTA,RPCS2(R0) ;SELECT PORT #A  
CLR RPDS1(R0) ;SEIZE THE DRIVE  
MOV #11,RPCS1(R0) ;ISSUE DRIVE CLEAR  
MOV #13,RPCS1(R0) ;RELEASE THE DRIVE  
MOVB PORTB,RPCS2(R0) ;SELECT PORT #B  
CLR RPDS1(R0) ;SEIZE THE DRIVE THROUGH PORT 'B'  
MOV #11,RPCS1(R0) ;ISSUE DRIVE CLEAR  
MOV #13,RPCS1(R0) ;RELEASE THE DRIVE  
MOVB PORTA,RPCS2(R0) ;SELECT PORT A  
MOV PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT  
MOV PORTA,SEIZPT ;SEIZING PORT'S ADDRESS  
MOV #3,RPCS1(R0) ;ISSUE AN UNLOAD COMMAND THROUGH PORT A  
TYPE ,SWTCHA ;SWITCH TO PORT A MESSAGE  
TYPE ,STANDBY ;TYPE PRESS STANDBY  
MOV PORTA,-(SP) ;SAVE PORTA FOR TYPEOUT  
;TYPE THE PORT NUMBER  
;GO TYPE--OCTAL ASCII  
;TYPE 1 DIGIT(S)  
;SUPPRESS LEADING ZEROS  
;CP-LF  
15: BIT #MOL,RPDS1(R0) ;TEST 'MOL'

# 805

```

2837 012662 001774          BEQ      15          ;BR IF NOT SET
2838 012664 012760 000011 000000    MOV      #11,RPCS1(RO) ;ISSUE A DRIVE CLEAR
2839 012672 012760 000021 000000    MOV      #21,RPCS1(RO) ;ISSUE A READIN PRESET
2840 012700 012760 000013 000000    MOV      #13,RPCS1(RO) ;RELEASE THE DRIVE
2841
2842          ;VERIFY THAT THE DRIVE IS IN NEUTRAL
2843
2844 012706 005037 001250          CLR      RELERR      ;CLEAR THE 'RELEASE ERROR' INDICATOR
2845 012712 012737 000012 001122    MOV      #RPS1,$BDAOR ;FORM THE ADDRESS OF RPS1 FOR TYPEOUT
2846 012720 060737 001122          ADD      RO,$BDAOR   ;ADD THE I/O BASE ADDRESS
2847 012724 012737 011600 001124    MOV      #MOL!PGM!DPR!DRY,$GDOAT ;COMPARISON CONSTANT
2848 012732 113760 001224 000010    MOVVB   PORTA,RPCS2(RO) ;SELECT PORT A.
2849 012740 016037 000012 001170    MOV      RPS1(RO),STMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
2850 012746 013737 011170 001164    MOV      STMP2,STMP0   ;COPY IT INTO 'STMP0'
2851 012754 042737 100100 001164    BIC      #ATA!VV,STMP0 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2852 012762 113760 001224 000010    MOVVB   PORTB,RPCS2(RO) ;SELECT PORT B.
2853 012770 016037 000012 001172    MOV      RPS1(RO),STMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
2854 012776 013737 001172 001166    MOV      STMP3,STMP1   ;COPY IT INTO 'STMP1'
2855 013004 042737 100100 001166    BIC      #ATA!VV,STMP1 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2856 013012 023737 001164 001166    CMP      STMP0,STMP1   ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2857 013020 001006          BNE      64$         ;BR . NOT
2858 013022 005737 001164          TST      STMP0        ;RE: 1TERS ARE THE SAME: ARE THEY ZERO ?
2859 013026 001037          BNE      66$         ;BR IF NOT
2860 013030 104034          ERROR    34         ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2861 013032 000137 013216          JMP      68$         ;BYPASS THE REST OF THE CHECKS
2862 013036 013737 001170 001126 64$:  MOV      STMP2,$BDDAT ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2863 013044 013737 001226 001234          MOV      PORTB,PTNBR  ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2864 013052 113760 001226 000010    MOVVB   PORTB,RPCS2(RO) ;SELECT PORT B.
2865 013060 005737 001164          TST      STMP0        ;SEE IF STATUS EQ 0 FROM PORT A.
2866 013064 001414          BEQ      65$         ;BR IF ZERO
2867 013066 013737 001224 001234    MOV      PORTA,PTNBR  ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2868 013074 013737 001172 001126    MOV      STMP3,$BDDAT ;'BAD DATA' FOR ERROR TYPE OUT
2869 013102 113760 001224 000010    MOVVB   PORTA,RPCS2(RO) ;SELECT PORT A.
2870 013110 005737 001166          TST      STMP1        ;SEE IF STATUS EQ ZERO FROM PORT B.
2871 013114 001004          BNE      66$         ;BR IF NOT
2872 013116 012737 177777 001250 65$:  MOV      #-1,RELERR   ;SET 'RELEASE ERROR' INDICATOR
2873 013124 104016          ERROR    16         ;TYPE ERROR MESSAGE 16
2874 013126 013737 001170 001126 66$:  MOV      STMP2,$BDDAT ;LOOK FOR BIT FAILURES WHEN RPS1 READ
2875 013134 013737 001224 001234          MOV      PORTA,PTNBR  ;CHANGE PORT NUMBER
2876 013142 042737 000100 001170    BIC      #VV,STMP2    ;DON'T CHECK THE VV BIT
2877 013150 023737 001124 001170    CMP      $GDOAT,STMP2 ;ALL BITS OK ?
2878 013156 001401          BEQ      67$         ;BR IF OK FROM PORT A.
2879 013160 104037          ERROR    37         ;REPORT ERROR
2880 013162 013737 001172 001126 67$:  MOV      STMP3,$BDDAT ;CHECK RPS1 FOR BIT FAILURES - FROM PORT B.
2881 013170 013737 001226 001234          MOV      PORTB,PTNBR  ;CHANGE PORT NUMBER
2882 013176 042737 000100 001172    BIC      #VV,STMP3    ;DON'T CHECK THE VV BIT
2883 013204 023737 001124 001172    CMP      $GDOAT,STMP3 ;SEE IF READ OK FROM PORT B.
2884 013212 001401          BEQ      68$         ;BR IF OK
2885 013214 104037          ERROR    37         ;REPORT THE ERROR
2886 013216 000240          NOP
2887
2888          ;IF ERROR OCCURRED, CHECK FOR LOOP ON TEST
2889
2890 013220 105737 001103          TSTB   SERFLG       ;DID AN ERROR OCCUR
2891 013224 001412          BEQ      25          ;BR IF NOT
2892 013226 032777 001000 165704    BIT      #SW09,$SWR   ;SEE IF LOOP ON ERROR (SWR9=1)

```



C05

```

2993 013234 001406 BEQ 25 ;BR IF NOT
2994 013236 105037 001103 CLR8 $ERFLG ;CLEAR THE ERROR FLAG
2995 013242 005037 001176 CLR $TIMES ;CLEAR THE MAX ITERATION COUNT
2996 013246 000177 165636 JMP $SLPERR ;GO TO THE LOOP ADDRESS
2997 013252 005737 001266 25: TST KYBCTL ;SINGLE TEST MODE ?
2998 013256 001406 BEQ 35 ;BR IF NOT
2999 013260 032777 040000 165652 BIT $SW14,$SWP ;LOOP ON TEST ?
3000 013266 001002 BNE 35 ;BR IF LOOPING
3001 013270 104401 024035 TYPE ,SWTCHN ;RETURN CONTROLLER SELECT SWITCH TO 'A/B'
3002 013274 35:
3003 013274 113760 001226 000010 MOV8 PORTB,$RPS2(RO) ;SELECT PORT B
3004 013302 013737 001226 001234 MOV PORTB,P'NBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
3005 013310 012760 000011 000000 MOV #11,$RPS1(RO) ;ISSUE A DRIVE CLEAR
3006 013316 012760 000021 000000 MOV #21,$RPS1(RO) ;ISSUE A READIN PRESET
3007 013324 012760 010000 000032 MOV #FMT22,$RPOF(RO) ;SET 'FMT22'
3008 013332 012760 000013 000000 MOV #13,$RPS1(RO) ;RELEASE PORT B
3009 013340 012737 011610 001254 MOV #5000,$WATCH ;SPINDLE MOTOR 'COOL DOWN' DELAY
3010 013346 005737 001254 45: TST WATCH ;FINISHED ?
3011 013352 001375 BNE 45 ;BR IF NOT
3012 013354 000004 SCOPE ;LOOP ?
3013 013356 000400 BR TST11 ;GO TO NEXT TEST
    
```

```

*****
*TEST 11 TEST 'CONTROLLER SELECT' SWITCH THROUGH PORT B DURING UNLOAD
*
*VERIFY THAT THE 'CONTROLLER SELECT' SWITCH IS INHIBITED WHEN THE
* RPO4 IS CYCLED DOWN BY AN UNLOAD COMMAND.
*
* A. ISSUE AN UNLOAD COMMAND THROUGH PORT B.
*
* B. REQUEST THAT THE OPERATOR SWITCH THE 'CONTROLLER SELECT' SWITCH
* TO B AND THEN PRESS THE 'STANDBY' BUTTON.
*
* C. WAIT FOR 'MOL' TO SET BY MONITORING RPS1 THROUGH PORT
* B. WHEN THE DRIVE HAS CYCLED UP ('MOL' HAS SET), ISSUE
* RELEASE COMMAND THROUGH PORT B.
*
* D. VERIFY THAT THE DRIVE CAN BE ACCESSED BY BOTH PORTS AND THAT
* THE DRIVE IS STILL IN NEUTRAL.
*
* E. REQUEST THAT THE OPERATOR RETURN THE 'CONTROLLER SELECT' SWITCH
* TO 'A/B'.
*****
    
```

```

3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037 013360 TST11: TST KYBCTL ;PERFORMING ONLY SINGLE TESTS ?
3038 013360 005737 001266 BEQ 25 ;BR IF NOT
3039 013364 001406 BPL 15 ;BR IF JUST ENTERED TEST
3040 013366 100002 JMP EXEC ;RETURN & GET NEXT TEST NUMBER
3041 013370 000137 002424 15: MOV #-1,KYBCTL ;SET SINGLE TEST INDICATOR
3042 013374 012737 177777 001266 25: MOV8 #11,$STNM ;TEST NUMBER
3043 013402 112737 000011 001102 MOV #TEST11,$LPADR ;LOAD LOOP ON TEST ADDRESS
3044 013410 012737 013432 001106 MOV #TEST11,$LPERR ;LOAD LOOP ON ERROR ADDRESS
3045 013416 012737 013432 001110 MOV #1,$TIMES ;DO 1 ITERATION
3046 013424 012737 000001 001176 TEST11: MOV #STACK,$P ;SETUP THE STACK POINTER
3047 013432 012706 001100
3048
    
```



# D05

MD-11-DZRJF-A, RPO4/5/6 DUAL CONTRLR LOGIC TEST - PART 2      MACY11 27(1006) 02-NOV-76 19:12 PAGE 55  
 DZRJFA.CMB      02-NOV-76 19:11      T11      TEST 'CONTROLLER SELECT' SWITCH THROUGH PORT B DURING UNLOAD

```

2949                                     ;CLEAR ATTENTION BITS FOR BOTH PORTS
2950
2951 013436 113760 001224 000010      MOVB   PORTA,RPCS2(R0) ;SELECT PORT #A
2952 013444 005060 000012              CLR    RPS1(R0)        ;SEIZE THE DRIVE
2953 013450 012760 000011 000000      MOV    #11,RPCS1(R0)  ;ISSUE DRIVE CLEAR
2954 013456 012760 000013 000000      MOV    #13,RPCS1(R0)  ;RELEASE THE DRIVE
2955 013464 113760 001226 000010      MOVB   PORTB,RPCS2(R0) ;SELECT PORT #B
2956 013472 005060 000012              CLR    RPS1(R0)        ;SEIZE THE DRIVE THROUGH PORT 'B'
2957 013476 012760 000011 000000      MOV    #11,RPCS1(R0)  ;ISSUE DRIVE CLEAR
2958 013504 012760 000013 000000      MOV    #13,RPCS1(R0)  ;RELEASE THE DRIVE
2959 013512 113760 001226 000010      MOVB   PORTB,RPCS2(R0) ;SELECT PORT B
2960 013520 013737 001226 001234      MOV    PORTB,PTNBR    ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2961 013526 013737 001226 001236      MOV    PORTB,SEIZPT   ;SEIZING PORT'S ADDRESS
2962 013534 012760 000003 000000      MOV    #3,RPCS1(R0)   ;ISSUE AN UNLOAD COMMAND THROUGH PORT B
2963 013542 104401 024207              TYPE   ,SWTCHB ;SWITCH TO PORT B MESSAGE
2964 013546 104401 023712              TYPE   ,STANDBY ;TYPE PRESS STANDBY
2965 013552 013746 001226              MOV    PORTB,-(SP)    ;SAVE PORTB FOR TYPEOUT
2966                                     ;TYPE THE PORT NUMBER
2967 013556 104403              TYPOS  ;GO TYPE--OCTAL ASCII
2968 013560 001                .BYTE  1 ;TYPE 1 DIGIT(S)
2969 013561 000                .BYTE  0 ;SUPPRESS LEADING ZEROS
2970 013562 104401 LJ1207              TYPE   ,$CRLF ;CR-LF
2971 013566 032760 010000 000012 1$:  BIT    #MOL,RPS1(R0)  ;TEST 'MOL'
2972 013574 001774              BEQ    1$ ;BR IF NOT SET
2973 013576 012760 000011 000000      MOV    #11,RPCS1(R0)  ;ISSUE A DRIVE CLEAR
2974 013604 012760 000021 000000      MOV    #21,RPCS1(R0)  ;ISSUE A READIN PRESET
2975 013612 012760 000013 000000      MOV    #13,RPCS1(R0)  ;RELEASE THE DRIVE
2976
2977                                     ;VERIFY THAT THE DRIVE IS IN NEUTRAL
2978
2979 013620 005037 001250              CLR    RELERR ;CLEAR THE 'RELEASE ERROR' INDICATOR
2980 013624 012737 000012 001122      MOV    #RPS1,$BODADR ;FORM THE ADDRESS OF RPS1 FOR TYPEOUT
2981 013632 060037 001122      ADD    R0,$BODADR    ;ADD THE I/O BASE ADDRESS
2982 013636 012737 011600 001124      MOV    #MOL!PGM!DPR!DRY,$CDDAT ;COMPARISON CONSTANT
2983 013644 113760 001224 000010      MOVB   PORTA,RPCS2(R0) ;SELECT PORT A.
2984 013652 016037 000012 001170      MOV    RPS1(R0),STMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
2985 013660 013737 001170 001164      MOV    STMP2,STMP0    ;COPY IT INTO 'STMP0'
2986 013666 042737 100100 001164      BIC    #ATA!VV,STMP0  ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2987 013674 113760 001226 000010      MOVB   PORTB,RPCS2(R0) ;SELECT PORT B.
2988 013702 016037 000012 001172      MOV    RPS1(R0),STMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
2989 013710 013737 001172 001166      MOV    STMP3,STMP1    ;COPY IT INTO 'STMP1'
2990 013716 042737 100100 001166      BIC    #ATA!VV,STMP1  ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2991 013724 023737 001164 001166      CMP    STMP0,STMP1    ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2992 013732 001006              BNE    64$ ;BR IF NOT
2993 013734 005737 001164              TST    STMP0 ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
2994 013740 001037              BNE    66$ ;BR IF NOT
2995 013742 104034              ERROR  34 ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2996 013744 000137 014130              JMP    68$ ;BYPASS THE REST OF THE CHECKS
2997 013750 013737 001170 001126 64$: MOV    STMP2,$BDDAT   ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2998 013756 013737 001226 001234      MOV    PORTB,PTNBR    ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2999 013764 113760 001226 000010      MOVB   PORTB,RPCS2(R0) ;SELECT PORT B.
3000 013772 005737 001164              TST    STMP0 ;SEE IF STATUS EQ 0 FROM PORT A.
3001 013776 001414              BEQ    65$ ;BR IF ZERO
3002 014000 013737 001224 001234      MOV    PORTA,PTNBR    ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
3003 014006 013737 001172 001126      MOV    STMP3,$BDDAT   ;'BAD DATA' FOR ERROR TYPE OUT
3004 014014 113760 001224 000010      MOVB   PORTA,RPCS2(R0) ;SELECT PORT A.

```

# E05

```

3005 014022 005737 001166          TST      $TMP1          ;SEE IF STATUS EG ZERO FROM PORT B.
3006 014026 001004          BNE      66$          ;BR IF NOT
3007 014030 012737 177777 001250 65$: MOV      #-1,RELERR   ;SET 'RELEASE ERROR' INDICATOR
3008 014036 104016          ERROR   16          ;TYPE ERROR MESSAGE 16
3009 014040 013737 001170 001126 66$: MOV      $TMP2,$BDDAT ;LOOK FOR BIT FAILURES WHEN RPO51 READ
3010 014046 013737 001224 001234          MOV      PORTA,PTNBR ;CHANGE PORT NUMBER
3011 014054 042737 000100 001170          BIC      #VV,$TMP2   ;DON'T CHECK THE VV BIT
3012 014062 023737 001124 001170          CMP      $GDDAT,$TMP2 ;ALL BITS OK ?
3013 014070 001401          BEQ      67$          ;BR IF OK FROM PORT A.
3014 014072 104037          ERROR   37          ;REPORT ERROR
3015 014074 013737 001172 001126 67$: MOV      $TMP3,$BDDAT ;CHECK RPO51 FOR BIT FAILURES - FROM PORT B.
3016 014102 013737 001226 001234          MOV      PORTB,PTNBR ;CHANGE PORT NUMBER
3017 014110 042737 000100 001172          BIC      #VV,$TMP3   ;DON'T CHECK THE VV BIT
3018 014116 023737 001124 001172          CMP      $GDDAT,$TMP3 ;SEE IF READ OK FROM PORT B.
3019 014124 001401          BEQ      68$          ;BR IF OK
3020 014126 104037          ERROR   37          ;REPORT THE ERROR
3021 014130 000240          68$:  NOP
3022
3023          ;IF ERROR OCCURRED, CHECK FOR LOOP ON TEST
3024
3025 014132 105737 001103          TSTB    $ERFLG       ;DID AN ERROR OCCUR
3026 014136 001412          BEQ      2$          ;BR IF NOT
3027 014140 032777 001000 164772          BIT     #SW09,$SWR   ;SEE IF LOOP ON ERROR (SWR9=1)
3028 014146 001406          BEQ      2$          ;BR IF NOT
3029 014150 105037 001103          CLRB    $ERFLG       ;CLEAR THE ERROR FLAG
3030 014154 005037 001176          CLR     $TIMES       ;CLEAR THE MAX ITERATION COUNT
3031 014160 000177 164724          JMP     @SLPERR      ;GO TO THE LOOP ADDRESS
3032 014164 005737 001266          2$:  TST     KYBCTL     ;SINGLE TEST MODE ?
3033 014170 001406          BEQ      3$          ;BR IF NOT
3034 014172 032777 040000 164740          BIT     #SW14,$SWR   ;LOOP ON TEST ?
3035 014200 001002          BNE     3$          ;BR IF LOOPING
3036 014202 104401 024035          TYPE    ,SWTCHN     ;RETURN CONTROLLER SELECT SWITCH TO 'A/B'
3037 014206
3038 014206 113760 001224 000010          3$:  MOVB    PORTA,RPCS2(RO) ;SELECT PORT A
3039 014214 013737 001224 001234          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
3040 014222 012760 000011 000000          MOV     #11,RPCS1(RO) ;ISSUE A DRIVE CLEAR
3041 014230 012760 000021 000000          MOV     #21,RPCS1(RO) ;ISSUE A READIN PRESET
3042 014236 012760 010000 000032          MOV     #FMT22,RPOF(RO) ;SET 'FMT22'
3043 014244 012760 000013 000000          MOV     #13,RPCS1(RO) ;RELEASE PORT A
3044 014252 012737 011610 001254          MOV     #5000.,WATCH ;SPINDLE MOTOR 'COOL DOWN' DELAY
3045 014260 005737 001254          4$:  TST     WATCH     ;FINISHED ?
3046 014264 001375          BNE     4$          ;BR IF NOT
3047 014266 000004          SCOPE
3048 014270 000400          BR     TST12        ;GO TO NEXT TEST
3049
3050
3051          ;*****
3052          *TEST 12      TEST 'CONTROLLER SELECT' SWITCH, DRIVE CYCLED UP
3053          *
3054          *TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED UP).
3055          *
3056          * A. SWITCH TO CONTROLLER 'A' POSITION. VERIFY THAT THE DRIVE IS IN
3057          * NEUTRAL AND THAT THE STATUS BITS IN RPO51, AS READ THROUGH BOTH
3058          * PORTS, ARE CORRECT.
3059          *
3060          * B. SWITCH TO CONTROLLER 'B' POSITION. VERIFY THAT THE DRIVE IS IN
  
```

# F05

3061  
3062  
3063  
3064  
3065  
3066  
3067  
3068  
3069  
3070  
3071  
3072  
3073  
3074  
3075  
3076  
3077  
3078  
3079  
3080  
3081  
3082  
3083  
3084  
3085  
3086  
3087  
3088  
3089  
3090  
3091  
3092  
3093  
3094  
3095  
3096  
3097  
3098  
3099  
3100  
3101  
3102  
3103  
3104  
3105  
3106  
3107  
3108  
3109  
3110  
3111  
3112  
3113  
3114  
3115  
3116

```

014272
014272 005737 001266
014276 001406
014300 100002
014302 000137 002424
014306 012737 177777 001266
014314 112737 000012 001102
014322 012737 014344 001106
014330 012737 014344 001110
014336 012737 000001 001176
014344 012706 001100

014350 113760 001224 000010
014356 005060 000012
014362 012760 000011 000000
014370 012760 000013 000000
014376 113760 001226 000010
014404 005060 000012
014410 012760 000011 000000
014416 012760 000013 000000
014424 104401 024124
014430 104401 024272
014434 000000

014436 005037 001250
014442 012737 000012 001122
014450 060037 001122
014454 012737 011700 001124
014462 113760 001224 000010
014470 016037 000012 001170
014476 013737 001170 001164
014504 042737 100100 001164
014512 113760 001226 000010
014520 016037 000012 001172
014526 013737 001172 001166
014534 042737 100100 001166
014542 023737 001164 001166
001006
014552 005737 001164
014556 001045
014560 104034
014562 000137 014746
014566 013737 001170 001126
014574 013737 001226 001234
014602 113760 001226 000010
  
```

```

; * NEUTRAL AND THAT THE STATUS BITS IN RPDS1, AS READ THROUGH BOTH
; * PORTS, ARE CORRECT.
; * C. RETURN THE 'CONTROLLER SELECT' SWITCH TO THE 'A/B' POSITION. VERIFY
; * THE DRIVE STATE.
;*****
TST12:
TST KYBCTL ;PERFORMING ONLY SINGLE TESTS ?
BEQ 25 ;BR IF NOT
BPL 15 ;BR IF JUST ENTERED TEST
JMP EXEC ;RETURN & GET NEXT TEST NUMBER
15: MOV #-1, KYBCTL ;SET SINGLE TEST INDICATOR
25: MOVB #12, $STNM ;TEST NUMBER
MOV #TEST12, $LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #TEST12, $LPERR ;LOAD LOOP ON ERROR ADDRESS
MOV #1, $TIMES ;DO 1 ITERATION
TEST12: MOV #STACK, SP ;SETUP THE STACK POINTER

;CLEAR ATTENTION BITS FOR BOTH PORTS
MOVB PORTA, RPCS2(RO) ;SELECT PORT #A
CLR RPDS1(RO) ;SEIZE THE DRIVE
MOV #11, RPCS1(RO) ;ISSUE DRIVE CLEAR
MOV #13, RPCS1(RO) ;RELEASE THE DRIVE
MOVB PORTB, RPCS2(RO) ;SELECT PORT #B
CLR RPDS1(RO) ;SEIZE THE DRIVE THROUGH PORT 'B'
MOV #11, RPCS1(RO) ;ISSUE DRIVE CLEAR
MOV #13, RPCS1(RO) ;RELEASE THE DRIVE
TYPE ,SWCHA ;SWITCH TO 'A'
TYPE ,CONTUE ;PRESS 'CONTINUE'
HALT

;VERIFY THAT THE DRIVE IS IN NEUTRAL
CLR RELERR ;CLEAR THE 'RELEASE ERROR' INDICATOR
MOV #RPDS1, $B0ADR ;FORM THE ADDRESS OF RPDS1 FOR TYPEOUT
ADD RO, $B0ADR ;ADD THE I/O BASE ADDRESS
MOV #M0L!PGM!DPR!DRY!VV, $C0DAT ;COMPARISON CONSTANT
MOVB PORTA, RPCS2(RO) ;SELECT PORT A.
MOV RPDS1(RO), $TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
MOV $TMP2, $TMP0 ;COPY IT INTO 'TMP0'
BIC #ATA!VV, $TMP0 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
MOVB PORTB, RPCS2(RO) ;SELECT PORT B.
MOV RPDS1(RO), $TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
MOV $TMP3, $TMP1 ;COPY IT INTO 'TMP1'
BIC #ATA!VV, $TMP1 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
CMP $TMP0, $TMP1 ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
BNE 645 ;BR IF NOT
TST $TMP0 ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
BNE 665 ;BR IF NOT
ERROR 34 ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
JMP 685 ;BYPASS THE REST OF THE CHECKS
645: MOV $TMP2, $B0DAT ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
MOV PORTB, PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
MOVB PORTB, RPCS2(RO) ;SELECT PORT B.
  
```

# G05

3117	014610	005737	001164			TST	\$TMP0	;SEE IF STATUS EQ 0 FROM PORT A.
3118	014614	001414				BEQ	65\$	;BR IF ZERO
3119	014616	013737	001224	001234		MOV	PORTA,PTNBR	;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
3120	014624	013737	001172	001126		MOV	\$TMP3,\$BDDAT	; 'BAD DATA' FOR ERROR TYPE OUT
3121	014632	113760	001224	000010		MOVB	PORTA,RPCS2(RO)	;SELECT PORT A.
3122	014640	005737	001166			TST	\$TMP1	;SEE IF STATUS EQ ZERO FROM PORT B.
3123	014644	001012				BNE	66\$	;BR IF NOT
3124	014646	012737	177777	001250	65\$:	MOV	#-1,RELERR	;SET 'RELEASE ERROR' INDICATOR
3125	014654	012760	000011	000000		MOV	#11,RPCS1(RO)	;CLEAR THE DRIVE
3126	014662	012760	000013	000000		MOV	#13,RPCS1(RO)	;RELEASE THE DRIVE
3127	014670	104017				ERROR	17	;TYPE ERROR MESSAGE 17
3128	014672	013737	001170	001126	66\$:	MOV	\$TMP2,\$BDDAT	;LOOK FOR BIT FAILURES WHEN RPOS1 READ
3129	014700	013737	001224	001234		MOV	PORTA,PTNBR	;CHANGE PORT NUMBER
3130	014706	023737	001124	001170		CMP	\$GDDAT,\$TMP2	;ALL BITS OK ?
3131	014714	001401				BEQ	67\$	;BR IF OK FROM PORT A.
3132	014716	104037				ERROR	37	;REPORT ERROR
3133	014720	013737	001172	001126	67\$:	MOV	\$TMP3,\$BDDAT	;CHECK RPOS1 FOR BIT FAILURES - FROM PORT B.
3134	014726	013737	001226	001234		MOV	PORTB,PTNBR	;CHANGE PORT NUMBER
3135	014734	023737	001124	001172		CMP	\$GDDAT,\$TMP3	;SEE IF READ OK FROM PORT B.
3136	014742	001401				BEQ	68\$	;BR IF OK
3137	014744	104037				ERROR	37	;REPORT THE ERROR
3138	014746	000240			68\$:	NOP		
3139	014750	104401	024207			TYPE	,SWTCHB	;SWITCH TO 'B'
3140	014754	104401	024272			TYPE	,CONTUE	;PRESS 'CONTINUE'
3141	014760	000000				HALT		
3142								
3143								;VERIFY THAT THE DRIVE IS IN NEUTRAL
3144								
3145	014762	005037	001250			CLR	RELERR	;CLEAR THE 'RELEASE ERROR' INDICATOR
3146	014766	012737	000012	001122		MOV	#RPOS1,\$BDDADR	;FORM THE ADDRESS OF RPOS1 FOR TYPEOUT
3147	014774	060037	001122			ADD	RO,\$BDDADR	;ADD THE I/O BASE ADDRESS
3148	015000	012737	011700	001124		MOV	#MOL!PGM!DPR!DRY!VV,\$GDDAT	;COMPARISON CONSTANT
3149	015006	113760	001224	000010		MOVB	PORTA,RPCS2(RO)	;SELECT PORT A.
3150	015014	016037	000012	001170		MOV	RPOS1(RO),\$TMP2	;GET THE DRIVE STATUS REGISTER FROM PORT A.
3151	015022	013737	001170	001164		MOV	\$TMP2,\$TMP0	;COPY IT INTO '\$TMP0'
3152	015030	042737	100100	001164		BIC	#ATA!VV,\$TMP0	;CLEAR PORT DEPENDENT BITS FROM THE COPY
3153	015036	113760	001226	000010		MOVB	PORTB,RPCS2(RO)	;SELECT PORT B.
3154	015044	016037	000012	001172		MOV	RPOS1(RO),\$TMP3	;GET THE DRIVE STATUS REGISTER FROM PORT B.
3155	015052	013737	001172	001166		MOV	\$TMP3,\$TMP1	;COPY IT INTO '\$TMP1'
3156	015060	042737	100100	001166		BIC	#ATA!VV,\$TMP1	;CLEAR PORT DEPENDENT BITS FROM THE COPY
3157	015066	023737	001164	001166		CMP	\$TMP0,\$TMP1	;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
3158	015074	001006				BNE	69\$	;BR IF NOT
3159	015076	005737	001164			TST	\$TMP0	;REGISTERS ARE THE SAME: ARE THEY ZERO ?
3160	015102	001045				BNE	71\$	;BR IF NOT
3161	015104	104034				ERROR	34	;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
3162	015106	000137	015272			JMP	73\$	;BYPASS THE REST OF THE CHECKS
3163	015112	013737	001170	001126	69\$:	MOV	\$TMP2,\$BDDAT	;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
3164	015120	013737	001226	001234		MOV	PORTB,PTNBR	;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
3165	015126	113760	001226	000010		MOVB	PORTB,RPCS2(RO)	;SELECT PORT B.
3166	015134	005737	001164			TST	\$TMP0	;SEE IF STATUS EQ 0 FROM PORT A.
3167	015140	001414				BEQ	70\$	;BR IF ZERO
3168	015142	013737	001224	001234		MOV	PORTA,PTNBR	;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
3169	015150	013737	001172	001126		MOV	\$TMP3,\$BDDAT	; 'BAD DATA' FOR ERROR TYPE OUT
3170	015156	113760	001224	000010		MOVB	PORTA,RPCS2(RO)	;SELECT PORT A.
3171	015164	005737	001166			TST	\$TMP1	;SEE IF STATUS EQ ZERO FROM PORT B.
3172	015170	001012				BNE	71\$	;BR IF NOT

H05

```

3173 015172 012737 177777 001250 70S: MOV #1,RELEA ;SET 'RELEASE ERROR' INDICATOR
3174 015200 012760 000011 000000 MOV #11,RPCS1(RO) ;CLEAR THE DRIVE
3175 015206 012760 000013 000000 MOV #13,RPCS1(RO) ;RELEASE THE DRIVE
3176 015214 104020 ERROR 20 ;TYPE ERROR MESSAGE 20
3177 015216 013737 001170 001126 71S: MOV $TMP2,$BDDAT ;LOOK FOR BIT FAILURES WHEN RPOSI READ
3178 015224 013737 001224 001234 MOV PORTA,PTNAR ;CHANGE PORT NUMBER
3179 015232 023737 001124 001170 CMP $GDDAT,$TMP2 ;ALL BITS OK ?
3180 015240 001401 BEQ 72S ;BR IF OK FROM PORT A.
3181 015242 104037 ERROR 37 ;REPORT ERROR
3182 015244 013737 001172 001126 72S: MOV $TMP3,$BDDAT ;CHECK RPOSI FOR BIT FAILURES - FROM PORT B.
3183 015252 013737 001226 001234 MOV PORTB,PTNBR ;CHANGE PORT NUMBER
3184 015260 023737 001124 001172 CMP $GDDAT,$TMP3 ;SEE IF READ OK FROM PORT B.
3185 015266 001401 BEQ 73S ;BR IF OK
3186 015270 104037 ERROR 37 ;REPORT THE ERROR
3187 015272 000240 73S: NOP
3188 015274 005737 001266 TST KYBCTL ;SINGLE TEST MODE ?
3189 015300 001402 BEQ 1S ;BR IF NOT
3190 015302 104401 TYPE ,SWTCHN ;RETURN SWITCH TO 'A/B'
3191 015306 000004 1S: SCOPE ;LOOP ?

```

```

*****
*TEST 13 TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT A
*
*TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED DOWN).
*
* A. CYCLE THE DRIVE DOWN.
*
* B. SWITCH TO CONTROLLER A POSITION. VERIFY THAT THE DRIVE IS IN
* NEUTRAL AND THAT THE STATUS BITS IN RPOSI, AS READ THROUGH BOTH
* PORTS, ARE CORRECT.
*
* C. SWITCH THE 'CONTROLLER SELECT' SWITCH TO A; CYCLE THE DRIVE UP.
*
* D. WHEN THE DRIVE CYCLES UP, VERIFY THAT 'VV-A IS RESET, AND
* THAT 'ATA-A IS SET.
*
* E. ISSUE A DRIVE CLEAR COMMAND AND A READIN PRESET COMMAND THROUGH
* PORT A.
*
* F. VERIFY THAT THE DRIVE CANNOT BE ACCESSED THROUGH PORT B AND
* 'NED' DOES NOT SET WHEN ATTEMPTING TO ACCESS THE DRIVE THROUGH
* PORT B. ATTEMPT TO SET PORT REQUEST BY WRITING 0'S
* INTO RPOSI THROUGH PORT B.
*
* G. ISSUE A RELEASE COMMAND THROUGH PORT A. VERIFY THAT THE
* DRIVE REMAINS LOCKED ON PORT A.

```

```

3223 015310
3224 015310 005737 001266 TST KYBCTL ;PERFORMING ONLY SINGLE TESTS ?
3225 015314 001406 BEQ 2S ;BR IF NOT
3226 015316 100002 BPL 1S ;BR IF JUST ENTERED TEST
3227 015320 000137 002424 JMP EXEC ;RETURN & GET NEXT TEST NUMBER
3228 015324 012737 177777 001266 1S: MOV #1,KYBCTL ;SET SINGLE TEST INDICATOR

```

```

3229 015332 112737 000013 001102 25:   MOVB   #13,STSTNM   ;TEST NUMBER
3230 015340 012737 015362 001106   MOV   #TEST13,$LPAOR ;LOAD LOOP ON TEST ADDRESS
3231 015346 012737 015362 001110   MOV   #TEST13,$LPERR ;LOAD LOOP ON ERROR ADDRESS
3232 015354 012737 000001 001176   MOV   #1,STIMES     ;DO 1 ITERATION
3233 015362 012706 001100   TEST13: MOV  #STACK,$P   ;SETUP THE STACK POINTER
3234 015366 113760 001224 000010   MOVB  PORTA,RPCS2(RO) ;SELECT PORT A
3235 015374 013737 001224 001234   MOV   PORTA,PTNBR   ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
3236 015402 104401 024341   TYPE  ,CYCLED      ;'CYCLE DOWN THE DRIVE'

```

\*\*\*\*\*  
;WAIT FOR 'MOL' TO RESET

```

3241 015406 032760 010000 000012 15:   BIT    #MOL,RPDS1(RO) ;TEST 'MOL'
3242 015414 001374   BNE   IS           ;BR IF IT IS STILL SET
3243 015416 104401 024124   TYPE  ,SWTCH      ;SWITCH TO 'A'
3244 015422 104401 024363   TYPE  ,CYCLEU     ;'CYCLE UP THE DRIVE'

```

\*\*\*\*\*  
;WAIT FOR DRIVE TO CYCLE UP AFTER SWITCH CHANGED

```

3249 015426 032760 010000 000012 25:   BIT    #MOL,RPDS1(RO) ;TEST 'MOL' AGAIN
3250 015434 001774   BEQ   25          ;BR IF NOT SET

```

\*\*\*\*\*  
;DRIVE IS CYCLED UP, CHECK STATUS THROUGH PORT A

```

3255 015436 005037 001244   CLR   CKERR       ;CLEAR THE 'CHECK ERROR' INDICATOR
3256 015442 016037 000012 001126   MOV   RPDS1(RO),SBDAT ;GET CONTENTS OF RPDS1
3257 015450 012737 000012 001122   MOV   #RPDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
3258 015456 060037 001122   ADD   RO,$BDADR     ;ADD RHI1 BASE ADDRESS
3259 015462 012737 110600 001124   MOV   #ATA!MOL!DPR!DRY,$GDDAT ;WHAT REGISTER SHOULD BE
3260 015470 013737 001126 001164   MOV   $BDAT,$STMP0  ;MOVE REGISTER CONTENTS TO 'STMP0'
3261 015476 042737 066077 001164   BIC   #111700,$STMP0 ;SAVE SPECIFIED BITS
3262 015504 023737 001124 001164   CMP   $GDDAT,$STMP0 ;COMPARE THE BITS
3263 015512 001414   BEQ   64$         ;BR IF OK
3264 015514 013737 001126 001174   MOV   $BDAT,$STMP4  ;COPY 'BAD DATA'
3265 015522 042737 111700 001174   BIC   #111700,$STMP4 ;CLEAR THE MASKED BITS
3266 015530 053737 001174 001124   BIS   $STMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
3267 015536 104021   ERROR 21         ;TYPE MESSAGE 21
3268 015540 005137 001244   COM   CKERR        ;SET THE REGISTER COMPARE ERROR INDICATOR
3269 015544 000240 64$:   NOP

```

\*\*\*\*\*  
;SET VOLUME VALID FOR PORT A

```

3274 015546 012760 000011 000000   MOV   #11,RPCS1(RO) ;ISSUE A DRIVE CLEAR
3275 015554 012760 000021 000000   MOV   #21,RPCS1(RO) ;ISSUE A READIN PRESET
3276 015562 012760 010000 000032   MOV   #FMT22,RPOF(RO) ;SET FMT22

```

\*\*\*\*\*  
;CHECK THE DRIVE STATUS THROUGH PORT B; VERIFY THAT 'MED' DOES NOT  
; SET WHEN THE DRIVE IS ACCESSED THROUGH PORT B.

```

3282 015570 113760 001226 000010   MOVB  PORTB,RPCS2(RO) ;SELECT PORT B
3283 015576 013737 001226 001234   MOV   PORTB,PTNBR   ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
3284 015604 005037 001244   CLR   CKERR        ;CLEAR THE 'CHECK ERROR' INDICATOR

```



# J05

MD-11-DZRJF-A, RPO4/5/6 DUAL CONTRLR LOGIC TEST - PART 2  
DZRJFA.CMB 02-NOV-76 19:11 T13

MAY11 27(1006) 02-NOV-76 19:12 PAGE 61  
TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT A

```
3285 015610 016037 000012 001126 MOV RPDS1(RO), $BDDAT ;GET CONTENTS OF RPDS1
3286 015616 012737 000012 001122 MOV #RPDS1, $BDAOR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
3287 015624 060037 001122 ADD RO, $BDAOR ;ADD RH11 BASE ADDRESS
3288 015630 005037 001124 CLR $GDDAT ;WHAT REGISTER SHOULD BE
3289 015634 013737 001126 001164 MOV $BDDAT, $TMP0 ;MOVE REGISTER CONTENTS TO 'STMP0'
3290 015642 042737 000077 001164 BIC #1C177700, $TMP0 ;SAVE SPECIFIED BITS
3291 015650 023737 001124 001164 CMP $GDDAT, $TMP0 ;COMPARE THE BITS
3292 015656 001414 BEQ 66$ ;BR IF OK
3293 015660 013737 001126 001174 MOV $BDDAT, $TMP4 ;COPY 'BAD DATA'
3294 015666 042737 177700 001174 BIC #177700, $TMP4 ;CLEAR THE MASKED BITS
3295 015674 053737 001174 001124 BIS $TMP4, $GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
3296 015702 104022 ERROR 22 ;TYPE MESSAGE 22
3297 015704 005137 001244 COM CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
3298 015710 000240 66$: NOP
3299 015712 005037 001244 CLR CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
3300 015716 016037 000010 001126 MOV RPCS2(RO), $BDDAT ;GET CONTENTS OF RPCS2
3301 015724 012737 000010 001122 MOV #RPCS2, $BDAOR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
3302 015732 060037 001122 ADD RO, $BDAOR ;ADD RH11 BASE ADDRESS
3303 015736 005037 001124 CLR $GDDAT ;WHAT REGISTER SHOULD BE
3304 015742 013737 001126 001164 MOV $BDDAT, $TMP0 ;MOVE REGISTER CONTENTS TO 'STMP0'
3305 015750 042737 167777 001164 BIC #1CNE0, $TMP0 ;SAVE SPECIFIED BITS
3306 015756 023737 001124 001164 CMP $GDDAT, $TMP0 ;COMPARE THE BITS
3307 015764 001414 BEQ 68$ ;BR IF OK
3308 015766 013737 001126 001174 MOV $BDDAT, $TMP4 ;COPY 'BAD DATA'
3309 015774 042737 010000 001174 BIC #NE0, $TMP4 ;CLEAR THE MASKED BITS
3310 016002 053737 001174 001124 BIS $TMP4, $GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
3311 016010 104023 ERROR 23 ;TYPE MESSAGE 23
3312 016012 005137 001244 COM CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
3313 016016 000240 68$: NOP
3314 016020 005060 000012 CLR RPDS1(RO) ;TRY TO SET REQUEST BY WRITING THROUGH
3315 ;THE LOCKED OUT PORT (PORT 'B')
3316
3317 ;*****
3318 ;VERIFY THAT DRIVE STAYS LOCKED ON PORT A
3319
3320 016024 113760 001224 000010 MOVB PORTA, RPCS2(RO) ;SELECT PORT A
3321 016032 013737 001224 001234 MOV PORTA, PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
3322 016040 012760 000013 000012 MOV #13, RPDS1(RO) ;ISSUE A RELEASE THROUGH PORT A
3323 016046 013737 001224 001236 MOV PORTA, SEIZPT ;ADDRESS OF 'LOCKED ON' PORT
3324 016054 113760 001226 000010 MOVB PORTB, RPCS2(RO) ;SELECT PORT B
3325 016062 013737 001226 001234 MOV PORTB, PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
3326 016070 005037 001244 CLR CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
3327 016074 016037 000012 001126 MOV RPDS1(RO), $BDDAT ;GET CONTENTS OF RPDS1
3328 016102 012737 000012 001122 MOV #RPDS1, $BDAOR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
3329 016110 060037 001122 ADD RO, $BDAOR ;ADD RH11 BASE ADDRESS
3330 016114 005037 001124 CLR $GDDAT ;WHAT REGISTER SHOULD BE
3331 016120 013737 001126 001164 MOV $BDDAT, $TMP0 ;MOVE REGISTER CONTENTS TO 'STMP0'
3332 016126 042737 000077 001164 BIC #1C177700, $TMP0 ;SAVE SPECIFIED BITS
3333 016134 023737 001124 001164 CMP $GDDAT, $TMP0 ;COMPARE THE BITS
3334 016142 001414 BEQ 70$ ;BR IF OK
3335 016144 013737 001126 001174 MOV $BDDAT, $TMP4 ;COPY 'BAD DATA'
3336 016152 042737 177700 001174 BIC #177700, $TMP4 ;CLEAR THE MASKED BITS
3337 016160 053737 001174 001124 BIS $TMP4, $GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
3338 016166 104024 ERROR 24 ;TYPE MESSAGE 24
3339 016170 005137 001244 COM CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
3340 016174 000240 70$: NOP
```



# K05

```
3341  
3342 ;IF ERROR OCCURRED, CHECK FOR LOOP ON TEST  
3343  
3344 016176 105737 001103 TSTB SERFLG ;DID AN ERROR OCCUR  
3345 016202 001412 BEQ 3$ ;BR IF NOT  
3346 016204 032777 001000 162726 BIT #SW09,2SWR ;SEE IF LOOP ON ERROR (SWR5=1)  
3347 016212 001406 BEQ 3$ ;BR IF NOT  
3348 016214 105037 001103 CLR8 SERFLG ;CLEAR THE ERROR FLAG  
3349 016220 005037 001176 CLR $TIMES ;CLEAR THE MAX ITERATION COUNT  
3350 016224 000177 162660 JMP 2SLPERR ;GO TO THE LOOP ADDRESS  
3351 016230 005737 001266 3$: TST KYBCTL ;IN SINGLE TEST MODE ?  
3352 016234 001460 BEQ 6$ ;BR IF NOT  
3353 016236 032777 040000 162674 BIT #SW14,2SWR ;LOOP ON TEST ?  
3354 016244 001054 BNE 6$ ;BR IF LOOPING  
3355 016246 104401 024341 TYPE ,CYCLED ;TYPE 'CYCLE DOWN'  
3356 016252 104401 024035 TYPE ,SWTCHN ;'SWITCH TO A/B'  
3357 016256 104401 024363 TYPE ,CYCLEU ;'CYCLE THE DRIVE UP'  
3358 016262 113760 001224 000010 MOV8 PORTA,RPCS2(RO) ;SELECT PORT A  
3359 016270 013737 001224 001234 MOV PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT  
3360 016276 032760 010000 000012 4$: BIT #MOL,RPDS1(RO) ;CHECK 'MOL'  
3361 016304 001374 BNE 4$ ;BR IF SET (DRIVE NOT CYCLED DOWN)  
3362 016306 032760 010000 000012 5$: BIT #MOL,RPDS1(RO) ;CHECK 'MOL' AGAIN  
3363 016314 001774 BEQ 5$ ;BR IF NOT SET (DRIVE NOT CYCLED UP)
```

```
3364  
3365 ;*****  
3366 ;SET VOLUME VALID FOR BOTH PORTS  
3367  
3368 016316 012760 000011 000000 MOV #11,RPCS1(RO) ;ISSUE A DRIVE CLEAR THROUGH PORT A  
3369 016324 012760 000021 000000 MOV #21,RPCS1(RO) ;ISSUE A READIN PRESET THROUGH PORT A  
3370 016332 012760 000013 000000 MOV #13,RPCS1(RO) ;RELEASE PORT A  
3371 016340 113760 001226 000010 MOV8 PORTB,RPCS2(RO) ;SELECT PORT B  
3372 016346 013737 001226 001234 MOV PORTB,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT  
3373 016354 012760 000021 000000 MOV #21,RPCS1(RO) ;ISSUE A READIN PRESET THROUGH PORT B  
3374 016362 012760 010000 000032 MOV #FMT22,RPOF(RO) ;SET FMT22  
3375 016370 012760 000013 000000 MOV #13,RPCS1(RO) ;RELEASE PORT B  
3376 016376 104401 001207 6$: TYPE ,SCRLF ;CR-LF  
3377 016402 012737 011610 001254 MOV #5000.,WATCH ;SPINDLE MOTOR 'COOL DOWN' DELAY  
3378 016410 005737 001254 7$: TST WATCH ;FINISHED ?  
3379 016414 001375 BNE 7$ ;BR IF NOT  
3380 016416 000004 SCOPE ;LOOP ?  
3381 016420 000400 BR TST14 ;GO TO NEXT TEST
```

```
3382  
3383 ;*****  
3384 *TEST 14 TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT B  
3385 *  
3386 *TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED DOWN).  
3387 *  
3388 * A. CYCLE THE DRIVE DOWN.  
3389 *  
3390 * B. SWITCH TO CONTROLLER B POSITION. VERIFY THAT THE DRIVE IS IN  
3391 * NEUTRAL AND THAT THE STATUS BITS IN RPDS1, AS READ THROUGH BOTH  
3392 * PORTS, ARE CORRECT.  
3393 *  
3394 * C. SWITCH THE 'CONTROLLER SELECT' SWITCH TO B; CYCLE THE DRIVE UP.  
3395 *  
3396 * D. WHEN THE DRIVE CYCLES UP, VERIFY THAT 'VV-B IS RESET, AND
```

L05

3397  
3398  
3399  
3400  
3401  
3402  
3403  
3404  
3405  
3406  
3407  
3408  
3409  
3410  
3411  
3412  
3413  
3414  
3415  
3416  
3417  
3418  
3419  
3420  
3421  
3422  
3423  
3424  
3425  
3426  
3427  
3428  
3429  
3430  
3431  
3432  
3433  
3434  
3435  
3436  
3437  
3438  
3439  
3440  
3441  
3442  
3443  
3444  
3445  
3446  
3447  
3448  
3449  
3450  
3451  
3452

016422  
016422 005737 001266  
016426 001406  
016430 100002  
016432 000137 002424  
016436 012737 177777 001266  
016444 112737 000014 001102  
016452 012737 016474 001106  
016460 012737 016474 001110  
016466 012737 000001 001176  
016474 012706 001100  
016500 113760 001226 000010  
016506 013737 001226 001234  
016514 104401 024341  
  
016520 032760 010000 000012  
016526 001374  
016530 104401 024207  
016534 104401 024363  
  
016540 032760 010000 000012  
016546 001774  
  
016550 005037 001244  
016554 016037 000012 001126  
016562 012737 000012 001122  
016570 060037 001122

- \* THAT 'ATA-B IS SET.
- \* E. ISSUE A DRIVE CLEAR COMMAND AND A READIN PRESET COMMAND THROUGH PORT B.
- \* F. VERIFY THAT THE DRIVE CANNOT BE ACCESSED THROUGH PORT A AND 'MED' DOES NOT SET WHEN ATTEMPTING TO ACCESS THE DRIVE THROUGH PORT A. ATTEMPT TO SET PORT REQUEST BY WRITING 0'S INTO RPDS1 THROUGH PORT A.
- \* G. ISSUE A RELEASE COMMAND THROUGH PORT B. VERIFY THAT THE DRIVE REMAINS LOCKED ON PORT B.
- \* H. CYCLE THE DRIVE DOWN. CHANGE THE 'CONTROLLER SELECT' SWITCH TO A/B; CYCLE THE DRIVE UP.
- \* I. VERIFY THAT BOTH PORTS CAN ACCESS THE DRIVE, THAT BOTH ATTENTION BITS ARE SET, AND THAT BOTH 'VV' BITS ARE RESET.

\*\*\*\*\*

```
TST14: TST KYBCTL ;PERFORMING ONLY SINGLE TESTS ?
        BEQ 2$ ;BR IF NOT
        BPL 1$ ;BR IF JUST ENTERED TEST
        JMP EXEC ;RETURN & GET NEXT TEST NUMBER
1$: MOV #-1, KYBCTL ;SET SINGLE TEST INDICATOR
2$: MOV #14, $STSTM ;TEST NUMBER
        MOV #TEST14, $LPADR ;LOAD LOOP ON TEST ADDRESS
        MOV #TEST14, $LPERR ;LOAD LOOP ON ERROR ADDRESS
        MOV #1, $TIMES ;DO 1 ITERATION
TEST14: MOV #STACK, SP ;SETUP THE STACK POINTER
        MOV# PORTB, RPCS2(RO) ;SELECT PORT B
        MOV PORTB, PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
        TYPE ,CYCLED ;'CYCLE DOWN THE DRIVE'
```

\*\*\*\*\*

;WAIT FOR 'MOL' TO RESET

```
1$: BIT #MOL, RPDS1(RO) ;TEST 'MOL'
      BNE 1$ ;BR IF IT IS STILL SET
      TYPE ,SWTCHB ;SWITCH TO 'B'
      TYPE ,CYCLEU ;'CYCLE UP THE DRIVE'
```

\*\*\*\*\*

;WAIT FOR DRIVE TO CYCLE UP AFTER SWITCH CHANGED

```
2$: BIT #MOL, RPDS1(RO) ;TEST 'MOL' AGAIN
      BEQ 2$ ;BR IF NOT SET
```

\*\*\*\*\*

;DRIVE IS CYCLED UP, CHECK STATUS THROUGH PORT B

```
CLR CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
MOV RPDS1(RO), $BDOAT ;GET CONTENTS OF RPDS1
MOV #RPDS1, $BDOADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
ADD RO, $BDOADR ;ADD RH11 BASE ADDRESS
```

# M05

```

3453 016574 012737 110600 001124      MOV      #ATA!MOL!DPR!DRY,$GDDAT ;WHAT REGISTER SHOULD BE
3454 016602 013737 001126 001164      MOV      $BDDAT,$TMP0 ;MOVE REGISTER CONTENTS TO 'TMP0'
3455 016610 042737 066077 001164      BIC      #1C111700,$TMP0 ;SAVE SPECIFIED BITS
3456 016616 023737 001124 001164      CMP      $GDDAT,$TMP0 ;COMPARE THE BITS
3457 016624 001414                BEQ      64$ ;BR IF OK
3458 016626 013737 001126 001174      MOV      $BDDAT,$TMP4 ;COPY 'BAD DATA'
3459 016634 042737 111700 001174      BIC      #111700,$TMP4 ;CLEAR THE MASKED BITS
3460 016642 053737 001174 001124      BIS      $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
3461 016650 104021                ERROR   21 ;TYPE MESSAGE 21
3462 016652 005137 001244      COM      CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
3463 016656 000240                64$:   NOP
3464
3465 ;*****
3466 ;SET VOLUME VALID FOR PORT B
3467
3468 016660 012760 000011 000000      MOV      #11,RPCS1(RO) ;ISSUE A DRIVE CLEAR
3469 016666 012760 000021 000000      MOV      #21,RPCS1(RO) ;ISSUE A READIN PRESET
3470 016674 012760 010000 000032      MOV      #FMT22,RPOF(RO) ;SET FMT22
3471
3472 ;*****
3473 ;CHECK THE DRIVE STATUS THROUGH PORT A; VERIFY THAT 'MED' DOES NOT
3474 ;SET WHEN THE DRIVE IS ACCESSED THROUGH PORT A.
3475
3476 016702 113760 001224 000010      MOVSB   PORTA,RPCS2(RO) ;SELECT PORT A
3477 016710 013737 001224 001234      MOV      PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
3478 016716 005037 001244                CLR      CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
3479 016722 016037 000012 001126      MOV      RPCS1(RO),$BDDAT ;GET CONTENTS OF RPCS1
3480 016730 012737 000012 001122      MOV      #RPCS1,$BDAOR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
3481 016736 060037 001122                ADD      RO,$BDAOR ;ADD RH11 BASE ADDRESS
3482 016742 005037 001124                CLR      $GDDAT ;WHAT REGISTER SHOULD BE
3483 016746 013737 001126 001164      MOV      $BDDAT,$TMP0 ;MOVE REGISTER CONTENTS TO 'TMP0'
3484 016754 042737 000077 001164      BIC      #1C177700,$TMP0 ;SAVE SPECIFIED BITS
3485 016762 023737 001124 001164      CMP      $GDDAT,$TMP0 ;COMPARE THE BITS
3486 016770 001414                BEQ      66$ ;BR IF OK
3487 016772 013737 001126 001174      MOV      $BDDAT,$TMP4 ;COPY 'BAD DATA'
3488 017000 042737 177700 001174      BIC      #177700,$TMP4 ;CLEAR THE MASKED BITS
3489 017006 053737 001174 001124      BIS      $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
3490 017014 104022                ERROR   22 ;TYPE MESSAGE 22
3491 017016 005137 001244      COM      CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
3492 017022 000240                66$:   NOP
3493 017024 005037 001244                CLR      CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
3494 017030 016037 000010 001126      MOV      RPCS2(RO),$BDDAT ;GET CONTENTS OF RPCS2
3495 017036 012737 000010 001122      MOV      #RPCS2,$BDAOR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
3496 017044 060037 001122                ADD      RO,$BDAOR ;ADD RH11 BASE ADDRESS
3497 017050 005037 001124                CLR      $GDDAT ;WHAT REGISTER SHOULD BE
3498 017054 013737 001126 001164      MOV      $BDDAT,$TMP0 ;MOVE REGISTER CONTENTS TO 'TMP0'
3499 017062 042737 167777 001164      BIC      #1CNED,$TMP0 ;SAVE SPECIFIED BITS
3500 017070 023737 001124 001164      CMP      $GDDAT,$TMP0 ;COMPARE THE BITS
3501 017076 001414                BEQ      68$ ;BR IF OK
3502 017100 013737 001126 001174      MOV      $BDDAT,$TMP4 ;COPY 'BAD DATA'
3503 017106 042737 010000 001174      BIC      #NED,$TMP4 ;CLEAR THE MASKED BITS
3504 017114 053737 001174 001124      BIS      $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
3505 017122 104023                ERROR   23 ;TYPE MESSAGE 23
3506 017124 005137 001244      COM      CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
3507 017130 000240                68$:   NOP
3508 017132 005060 000012      CLR      RPCS1(RO) ;TRY TO SET REQUEST BY WRITING THROUGH

```

N05

MD-11-DZRJF-A, RPO4/5/6 DUAL CONTRLR LOGIC TEST - PART 2  
DZRJFA.CMB 02-NOV-76 19:11

MACY11 27(1006) 02-NOV-76 19:12 PAGE 65  
T14 TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT B

```

3509                                     ;THE LOCKED OUT PORT (PORT 'A')
3510
3511                                     ;*****
3512                                     ;VERIFY THAT DRIVE STAYS LOCKED ON PORT B
3513
3514 017136 113760 001226 000010      MOVB   PORTB,RPCS2(RO) ;SELECT PORT B
3515 017144 013737 001226 001234      MOV    PORTB,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
3516 017152 012760 000013 000012      MOV    #13,RPDS1(RO) ;ISSUE A RELEASE THROUGH PORT B
3517 017160 013737 001226 001236      MOV    PORTB,SEIZPT ;ADDRESS OF 'LOCKED ON' PORT
3518 017166 113760 001224 000010      MOVB   F RTA,RPCS2(RO) ;SELECT PORT A
3519 017174 013737 001224 001234      MOV    PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
3520 017202 005037 001244                CLR    CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
3521 017206 016037 000012 001126      MOV    RPDS1(RO),%BDDAT ;GET CONTENTS OF RPDS1
3522 017214 012737 000012 001122      MOV    #RPDS1,%BDDADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
3523 017222 060037 001122                ADD    RO,%BDDADR ;ADD RH11 BASE ADDRESS
3524 017226 005037 001124                CLR    %GDDAT ;WHAT REGISTER SHOULD BE
3525 017232 013737 001126 001164      MOV    %BDDAT,%STMPD ;MOVE REGISTER CONTENTS TO 'STMPD'
3526 017240 042737 000077 001164      BIC    #1C17770C,%STMPD ;SAVE SPECIFIED BITS
3527 017246 023737 001124 001164      CMP    %GDDAT,%STMPD ;COMPARE THE BITS
3528 017254 001414                BFC    70S ;BR IF OK
3529 017256 013737 001126 001174      MOV    %BDDAT,%STMP4 ;COPY 'BDD' DATA
3530 017264 042737 177700 001174      BIC    #177700,%STMP4 ;CLEAR THE MASKED BITS
3531 017272 053737 001174 001124      BIS    %STMP4,%GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
3532 017300 104024                EORR   24 ;TYPE MESSAGE 24
3533 017302 005137 001244                COM    CKERR ;SET THE REGISTER COMPARE ERROR INDICATOR
3534 017306 000240                NOP
3535
3536                                     ;IF ERROR OCCURRED, CHECK FOR LOOP ON TEST
3537
3538 017310 105737 001103                TSTB   %ERFLG ;DID AN ERROR OCCUR
3539 017314 001412                BEQ    3S ;BR IF NOT
3540 017316 032777 001000 161614      BIT    #SW09,%SWR ;SEE IF LOOP ON ERROR (SWR9=1)
3541 017324 001406                BEQ    3S ;BR IF NOT
3542 017326 105037 001103                CLRB   %ERFLG ;CLEAR THE ERROR FLAG
3543 017332 005037 001176                CLR    %STIMES ;CLEAR THE MAX ITERATION COUNT
3544 017336 000177 161546                JMP    %SLPERR ;GO TO THE LOOP ADDRESS
3545 017342 032777 040000 161570 3S:  BIT    #SW14,%SWR ;LOOP ON TEST ?
3546 017350 001054                BNE    6S ;BR IF LOOPING
3547 017352 104401 024341                TYPE   ',CYCLED' ;TYPE 'CYCLE DOWN'
3548 017356 104401 024035                TYPE   ',SWTCHN' ;'SWITCH TO A/B'
3549 017362 104401 024363                TYPE   ',CYCLEU' ;'CYCLE THE DRIVE UP'
3550 017366 113760 001226 000010      MOVB   PORTB,RPCS2(RO) ;SELECT PORT B
3551 017374 013737 001226 001234      MOV    PORTB,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
3552 017402 032760 010000 000012 4S:  BIT    #MOL,RPDS1(RO) ;CHECK 'MOL'
3553 017410 001374                BNE    4S ;BR IF SET (DRIVE NOT CYCLED DOWN)
3554 017412 032760 010000 000012 5S:  BIT    #MOL,RPDS1(RO) ;CHECK 'MOL' AGAIN
3555 017420 001774                BEQ    5S ;BR IF NOT SET (DRIVE NOT CYCLED UP)
3556
3557                                     ;*****
3558                                     ;SET VOLUME VALID FOR BOTH PORTS
3559
3560 017422 012760 000011 000000      MOV    #11,RPCS1(RO) ;ISSUE A WRITE CLEAR THROUGH PORT B
3561 017430 012760 000021 000000      MOV    #21,RPCS1(RO) ;ISSUE A READIN PRESET THROUGH PORT B
3562 017436 012760 000013 000000      MOV    #13,RPCS1(RO) ;RELEASE PORT B
3563 017444 113760 001224 000010      MOVB   PORTA,RPCS2(RO) ;SELECT PORT A
3564 017452 013737 001224 001234      MOV    PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT

```

# 806

```

3565 017460 012760 000021 000000      MOV      #21,RPCS1(RO)  ;ISSUE A READIN PRESET THROUGH PORT A
3566 017466 012760 010000 000032      MOV      #FMT22,RPOF(RO) ;SET FMT22
3567 017474 012760 000013 000000      MOV      #13,RPCS1(RO)  ;RELEASE PORT A
3568 017502 012737 0116.0 001254 65:      MOV      #5000.,WATCH  ;SPINDLE MOTOR 'COOL DOWN' DELAY
3569 017510 005737 001254 75:      TST      WATCH         ;FINISHED ?
3570 017514 001375      BNE      75            ;BR IF NOT
3571 017516 000004      SCOPE                    ;LOOP ?
3572 017520 000137 017524      JMP      SEOP           ;GO TO THE END OF PASS ROUTINE
3573
3574      .SBTTL  END OF PASS ROUTINE
3575
3576      ;*****
3577      ;INCREMENT THE PASS NUMBER ($PASS)
3578      ;INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
3579      ;TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
3580      ;WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
3581      ;IF THERES A MONITOR GO TO IT
3582      ;IF THERE ISN'T JUMP TO TSTIAA
3583
3584      SEOP:
3585 017524 005737 001266      TST      KYBCTL        ;ENTERED TEST VIA KEYBOARD COMMAND ?
3586 017530 001402      BEQ      .+6           ;BR IF NOT
3587 017532 000137 002424      JMP      EXEC          ;RETURN TO KEYBOARD CONTROL
3588 017536 005037 001102      CLR      $TSTNM        ;ZERO THE TEST NUMBER
3589 017542 0051037 001176      CLR      $TIMES        ;ZERO THE NUMBER OF ITERATIONS
3590 017546 005237 001100      INC      $PASS         ;INCREMENT THE PASS NUMBER
3591 017552 042737 100000 001100      BIC      #100000,$PASS ;DON'T ALLOW A NEG. NUMBER
3592 017560 005327      DEC      (PC)+         ;LOOP?
3593 017562 000001      SEOPCT: .WORD      1
3594 017564 003063      BGT      $DOAGN        ;: YES
3595 017566 012737      MOV      (PC)+,(PC)+   ;: RESTORE COUNTER
3596 017570 000001      SENDCT: .WORD      1
3597 017572 017562      SEOPCT
3598 017574 104401 017602      TYPE     65$          ;: TYPE ASCIZ STRING
3599 017600 000407      BR       64$          ;: GET OVER THE ASCIZ
3600
3601 017620      ;: 65$: .ASCIZ <12><15>/END PASS #/
3602 017620 013746 001100 64$:      MOV      $PASS,-(SP)   ;: SAVE $PASS FOR TYPEOUT
3603      ;: TYPE PASS NUMBER
3604 017624 104405      TYPDS    ;: GO TYPE--DECIMAL ASCII WITH SIGN
3605 017626 104401 017634      TYPE     67$          ;: TYPE ASCIZ STRING
3606 017632 000421      BR       66$          ;: GET OVER THE ASCIZ
3607      ;: 67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
3608 017676 66$:
3609 017676 013746 001112      MOV      $ERTTL,-(SP)  ;: SAVE $ERTTL FOR TYPEOUT
3610      ;: TOTAL NUMBER OF ERRORS
3611 017702 104405      TYPDS    ;: GO TYPE--DECIMAL ASCII WITH SIGN
3612 017704 104401 001207      TYPE     $CARLF        ;: TYPE CARRIAGE RETURN, LINE FEED
3613 017710 005037 001112      CLR      $ERTTL        ;: CLEAR ERROR TOTAL
3614 017714 013700 000042  $GET42: MOV      #42,RO      ;: GET MONITOR ADDRESS
3615 017720 001405      BEQ      $DOAGN        ;: BRANCH IF NO MONITOR
3616 017722 000005      RESET    ;: CLEAR THE WORLD
3617 017724 004710  SENDAD: JSR      PC,(RO)    ;: GO TO MONITOR
3618 017726 000240      NOP      ;: SAVE ROOM
3619 017730 000240      NOP      ;: FOR
3620 017732 000240      NOP      ;: ACT!!
  
```

END OF PASS ROUTINE

```

3621 017734          SDOAGN:
3622 017734 000137          JMP      2(PC)+          ;;RETURN
3623 017736 002706          SRTNAD: .WORD  TST1AA
3624 017740          377      000  SENULL: .BYTE  -1,-1,0      ;;NULL CHARACTER STRING
3625          017744          .EVEN
3626
3627          ;;*****
3628
3629          .SBTTL  *** SUBROUTINES ***
3630
3631          ;;*****
3632
3633
3634          ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS
3635
3636 017744 012737 020014 000004 CKCLK:  MOV      @CKCLK1,@ERRVEC ;SET UP VECTOR FOR CLOCK CHECK
3637 017752 005037 000006          CLR      @ERRVEC+2      ;NEW PSW
3638 017756 005777 161230          TST     @SLKCSR        ;CHECK FOR KW11-P
3639 017762 013701 001216          MOV     @SLPVEC,R1     ;KW11-P VECTOR ADDRESS
3640 017766 012721 020076          MOV     @CLOCK,(R1)+  ;SET UP KW11-P VECTOR
3641 017772 012711 000300          MOV     #300,(R1)     ;PSW - PRI 6
3642 017776 012777 177777 161210          MOV     @-1,@SLKCSB   ;LOAD COUNTER BUFFER WITH 1'S
3643 020004 012777 000135 161200          MOV     @135,@SLKCSR  ;SET CLOCK - CNT UP, 16MS, CONT INT
3644 020012 000425          BR      CKCLK3
3645 020014 062706 000004          CKCLK1: ADD     #4,SP      ;RESTORE THE STACK POINTER
3646 020020 012737 020056 000004          MOV     @CKCLK2,@ERRVEC ;CHANGE ERROR VECTOR TO CHECK FOR KW11-L
3647 020026 005777 161166          TST     @SLKS         ;LOOK FOR KW11-L
3648 020032 013701 001222          MOV     @SLPVEC,R1     ;KW11-L VECTOR ADDRESS
3649 020036 012721 020076          MOV     @CLOCK,(R1)+  ;SET UP KW11-L VECTOR
3650 020042 012711 000300          MOV     #300,(R1)     ;PSW - PRI 6
3651 020046 012777 000100 161144          MOV     @100,@SLKS    ;SET KW11-L INTERRUPT
3652 020054 000404          BR      CKCLK3
3653 020056 062706 000004          CKCLK2: ADD     #4,SP      ;RESTORE THE STACK POINTER
3654 020062 062716 000002          ADD     #2,(SP)       ;INCREMENT RETURN, NO CLOCK
3655 020066 012737 000006 000004 CKCLK3: MOV     @6,@ERRVEC ;RESTORE THE ERROR VECTOR
3656 020074 000207          RTS      PC
3657
3658          ;ROUTINE TO COUNT CLOCK TICKS
3659
3660 020076 062737 000021 001252 CLOCK:  ADD     #17,TIME      ;ADD 17 MS TO ELAPSED TIME COUNTER
3661 020104 005737 001254          TST     WATCH         ;IS WATCH ALREADY ZERO ?
3662 020110 001406          BEQ     IS            ;BR IF IT IS
3663 020112 162737 000021 001254          SUB     #17,WATCH     ;SUBTRACT 17 MS FROM WATCH DOG COUNTER
3664 020120 100002          BPL     IS            ;BR IF NOT MINUS
3665 020122 005037 001254          CLR     WATCH        ;CLEAR WATCH DOG COUNTER
3666 020126 000002          IS:      RTI         ;RETURN
3667
3668          ;ROUTINE TO CALCULATE + 25% TIME TOLERANCE VALUES
3669
3670 020130 005746          TOLER: TST     -(SP)     ;MAKE ROOM ON THE STACK
3671 020132 016616 000002          MOV     2(SP),(SP)    ;SAVE STACK
3672 020136 013546          MOV     @2(R5)+,-(SP) ;GET TIME VALUE
3673 020140 011666 000004          MOV     (SP),4(SP)   ;MOVE TIME VALUE
3674 020144 006216          ASR     (SP)          ;DIVIDE BY 2
3675 020146 006216          ASR     (SP)          ;DIVIDE BY 2 AGAIN (FOR A TOTAL OF 4)
3676 020150 062666 000002          ADD     (SP)+,2(SP)  ;CALCULATE UPPER LIMIT FOR TIMEOUT

```



\*\*\* SUBROUTINES \*\*\*

3677 020154 000205 RTS R5 ;RETURN WITH TOLERANCES ON THE STACK

3678  
3679 ;;\*\*\*\*\*

3680  
3681 .SBTTL 'SYSMAC' UTILITY ROUTINES

3682  
3683 .SBTTL SCOPE HANDLER ROUTINE

3684  
3685 ;;\*\*\*\*\*

3686 ;\*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT

3687 ;\*AND LOAD THE TEST NUMBER(\$STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)

3688 ;\*AND LOAD THE ERROR FLAG (\$ERFLG) INTO DISPLAY<15:08>

3689 ;\*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:

3690 ;\*SW14=1 LOOP ON TEST

3691 ;\*SW11=1 INHIBIT ITERATIONS

3692 ;\*CALL

3693 ;\* SCOPE ;;SCOPE=IOT

3694  
3695 \$SCOPE:

3696 020156 104407  
3697 020160 032777 040000 160752 1\$: CKSWR ; TEST FOR CHANGE IN SOFT-SWR

3698 020166 001055 BIT #BIT14,2SWR ; LOOP ON PRESENT TEST?

3699 ; BNE \$OVER ; YES IF SW14=1

3700 020170 000416 ; \*\*\*\*\*START OF CODE FOR THE XOR TESTER\*\*\*\*\*

3701 \$XTSTR: BR 6\$ ; IF RUNNING ON THE "XOR" TESTER CHANGE

3702 020172 013746 000004 MOV @#ERRVEC, -(SP) ; THIS INSTRUCTION TO A "NOP" (NOP=240)

3703 020176 012737 020216 000004 MOV #5, @#ERRVEC ; SAVE THE CONTENTS OF THE ERROR VECTOR

3704 020204 005737 177060 TST @#177060 ; SET FOR TIMEOUT

3705 020210 012637 000004 MOV (SP)+, @#ERRVEC ; TIME OUT ON XOR?

3706 020214 000436 BR \$SVLAD ; RESTORE THE ERROR VECTOR

3707 020216 022626 5\$: CMP (SP)+, (SP)+ ; GO TO THE NEXT TEST

3708 020220 012637 000004 MOV (SP)+, @#ERRVEC ; CLEAR THE STACK AFTER A TIME OUT

3709 020224 000436 BR \$OVER ; RESTORE THE ERROR VECTOR

3710 020226 6\$; \*\*\*\*\*END OF CODE FOR THE XOR TESTER\*\*\*\*\*

3711 020226 105737 001103 2\$: TSTB \$ERFLG ; HAS AN ERROR OCCURRED?

3712 020232 001404 BEQ 3\$ ; BR IF NO

3713 020234 105037 001103 4\$: CLRB \$ERFLG ; ZERO THE ERROR FLAG

3714 020240 005037 001176 CLR \$TIMES ; CLEAR THE NUMBER OF ITERATIONS TO MAKE

3715 020244 032777 004000 160656 3\$: BIT #BIT11,2SWR ; INHIBIT ITERATIONS?

3716 020252 001011 BNE 1\$ ; BR IF YES

3717 020254 005737 001100 TST \$PASS ; IF FIRST PASS OF PROGRAM

3718 020260 001406 BEQ 1\$ ; INHIBIT ITERATIONS

3719 020262 005237 001104 INC \$ICNT ; INCREMENT ITERATION COUNT

3720 020266 023737 001176 001104 CMP \$TIMES, \$ICNT ; CHECK THE NUMBER OF ITERATIONS MADE

3721 020274 002012 BGE \$OVER ; BR IF MORE ITERATION REQUIRED

3722 020276 012737 000001 001104 1\$: MOV #1, \$ICNT ; REINITIALIZE THE ITERATION COUNTER

3723 020304 013737 020336 001176 MOV \$MXCNT, \$TIMES ; SET NUMBER OF ITERATIONS TO DO

3724 020312 105237 001102 \$SVLAD: INCB \$STNM ; COUNT TEST NUMBERS

3725 020316 011637 001106 MOV (SP), \$LPADR ; SAVE SCOPE LOOP ADDRESS

3726 020322 013777 001102 160612 \$OVER: MOV \$STNM, @DISPLAY ; DISPLAY TEST NUMBER

3727 020330 013716 001106 MOV \$LPADR, (SP) ; FUDGE RETURN ADDRESS

3728 020334 000002 RTI ; FIXES PS

3729 020336 000004 \$MXCNT: 4 ; MAX. NUMBER OF ITERATIONS

3730 .SBTTL ERROR HANDLER ROUTINE

3731 ;;\*\*\*\*\*

3732



# E06

```

3733 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3734 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3735 ;*AND GO TO $ERRTYP ON ERROR
3736 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3737 ;*SW15=1 HALT ON ERROR
3738 ;*SW13=1 INHIBIT ERROR TYPEOUTS
3739 ;*SW10=1 BELL ON ERROR
3740 ;*CALL
3741 ;* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3742
3743 $ERROR:
3744 020340 104407 001102 001242 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
3745 020342 113737 001102 001242 MOV $STNM,TSTNUM
3746 020350 105237 001103 75: INCB $ERFLG ;;SET THE ERROR FLAG
3747 020354 001775 BEQ 75 ;;DON'T LET THE FLAG GO TO ZERO
3748 020356 013777 001102 160556 MOV $STNM,$DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
3749 020364 032777 002000 160546 BIT $BIT10,$SWR ;;BELL ON ERROR?
3750 020372 001402 BEQ 15 ;;NO - SKIP
3751 020374 104401 001202 TYPE $BELL ;;RING BELL
3752 020400 005237 001112 15: INC $ERTTL ;;COUNT THE NUMBER OF ERRORS
3753 020404 011637 001116 MOV (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
3754 020410 162737 000002 001116 SUB #2,$ERRPC
3755 020416 117737 160474 001114 MOV $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
3756 020424 032777 020000 160506 BIT $BIT13,$SWR ;;SKIP TYPEOUT IF SET
3757 020432 001004 BNE 205 ;;SKIP TYPEOUTS
3758 020434 004737 020472 JSR PC,$ERRTYP ;;GO TO USER ERROR ROUTINE
3759 020440 104401 001207 TYPE ,SCLF
3760 020444
3761 020444 005777 160470 205: TST $SWR ;;HALT ON ERROR
3762 020450 100002 25: BPL 35 ;;SKIP IF CONTINUE
3763 020452 000000 HALT ;;HALT ON ERROR!
3764 020454 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
3765 020456
3766 020456 022737 017724 000042 35: CMP $SENDAD,$#42 ;;ACT-11 AUTO-ACCEPT?
3767 020464 001001 BNE 65 ;;BRANCH IF NO
3768 020466 000000 HALT ;;YES
3769 020470
3770 020470 000002 65: RTI ;;RETURN
3771 .SBTTL ERROR MESSAGE TYPEOUT ROUTINE
3772
3773 ;******
3774 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
3775 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
3776 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
3777
3778 $ERRTYP:
3779 020472 104401 001207 TYPE ,SCLF ;; "CARRIAGE RETURN" & "LINE FEED"
3780 020476 010046 MOV RO,-(SP) ;;SAVE RO
3781 020500 005000 CLR RO ;;PICKUP THE ITEM INDEX
3782 020502 153700 001114 BISB $ITEMB,RO
3783 020506 001004 BNE 15 ;;IF ITEM NUMBER IS ZERO, JUST
3784 3784 MOV $ERRPC,-(SP) ;;TYPE THE PC OF THE ERROR
3785 020510 013746 001116 ;;SAVE $ERRPC FOR TYPEOUT
3786 3786 ;;ERROR ADDRESS
3787 020514 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3788 020516 000445 BR 105 ;;GET OUT

```

```

3789 020520 005300      15:  DEC      RO      ;; ADJUST THE INDEX SO THAT IT WILL
3790 020522 006300      ASL      RO      ;; WORK FOR THE ERROR TABLE
3791 020524 006300      ASL      RO
3792 020526 006300      ASL      RO
3793 020530 062700 001276  ADD      #ERRTB,RO  ;; FORM TABLE POINTER
3794 020534 012037 020544  MOV      (RO)+,2$  ;; PICKUP "ERROR MESSAGE" POINTER
3795 020540 001404      BEQ      3$      ;; SKIP TYPEOUT IF NO POINTER
3796 020542 104401      TYPE     ;; TYPE THE "ERROR MESSAGE"
3797 020544 000000      25:  .WORD   0      ;; "ERROR MESSAGE" POINTER GOES HERE
3798 020546 104401 001207  TYPE     $CRLF  ;; "CARRIAGE RETURN" & "LINE FEED"
3799 020552 012037 020562  35:  MOV      (RO)+,4$  ;; PICKUP "DATA HEADER" POINTER
3800 020556 001404      BEQ      5$      ;; SKIP TYPEOUT IF 0
3801 020560 104401      TYPE     ;; TYPE THE "DATA HEADER"
3802 020562 000000      45:  .WORD   0      ;; "DATA HEADER" POINTER GOES HERE
3803 020564 104401 001207  TYPE     $CRLF  ;; "CARRIAGE RETURN" & "LINE FEED"
3804 020570 010146      55:  MOV      R1,-(SP)  ;; SAVE R1
3805 020572 012001      MOV      (RO)+,R1  ;; PICKUP "DATA TABLE" POINTER
3806 020574 001415      BEQ      9$      ;; BR IF NO DATA TO BE TYPED
3807 020576 012000      MOV      (RO)+,RO  ;; PICKUP "DATA FORMAT" POINTER
3808 020600 105720      65:  TSTB    (RO)+    ;; "OCTAL" OR "DECIMAL"
3809 020602 001003      BNE      7$      ;; BR IF DECIMAL
3810 020604 013146      MOV      @R1+,-(SP)  ;; SAVE @R1+ FOR TYPEOUT
3811 020606 104402      TYPOC   ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
3812 020610 000402      BR       8$
3813 020612      75:
3814 020612 013146      MOV      @R1+,-(SP)  ;; SAVE @R1+ FOR TYPEOUT
3815 020614 104405      TYPOS   ;; GO TYPE--DECIMAL ASCII WITH SIGN
3816 020616 005711      85:  TST      (R1)    ;; IS THERE ANOTHER NUMBER?
3817 020620 001403      BEQ      9$      ;; BR IF NO
3818 020622 104401 020642  TYPE     ,11$    ;; TYPE TWO(2) SPACES
3819 020626 000764      BR       6$      ;; LOOP
3820
3821 020630 012601      95:  MOV      (SP)+,R1  ;; RESTORE R1
3822 020632 012600      105: MOV      (SP)+,RO  ;; RESTORE RO
3823 020634 104401 001207  TYPE     $CRLF  ;; "CARRIAGE RETURN" & "LINE FEED"
3824 020640 000207      RTS     PC      ;; RETURN
3825 020642 020040 000      115: .ASCIZ  / /    ;; TWO(2) SPACES
3826 020646
3827 .SBTTL TYPE ROUTINE
3828
3829 *****
3830 *ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
3831 *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
3832 *NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
3833 *NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
3834 *NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
3835 *
3836 *CALL:
3837 *1) USING A TRAP INSTRUCTION
3838 *   TYPE ,MESADR ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
3839 *OR
3840 *   TYPE
3841 *   MESADR
3842 *
3843
3844 020646 105737 001157  $TYPE: TSTB  $TPFLG ;; IS THERE A TERMINAL?

```

```

3845 020652 100002      BPL      1$      ;; BR IF YES
3846 020654 000000      HALT      ;; HALT HERE IF NO TERMINAL
3847 020656 000407      BR      3$      ;; LEAVE
3848 020660 010046      1$: MOV    RO, -(SP)  ;; SAVE RO
3849 020662 017600 000002  MOV    2(SP), RO  ;; GET ADDRESS OF ASCIZ STRING
3850 020666 112046      2$: MOVB  (RO)+, -(SP)  ;; PUSH CHARACTER TO BE TYPED ONTO STACK
3851 020670 001005      BNE     4$      ;; BR IF IT ISN'T THE TERMINATOR
3852 020672 005726      TST    (SP)+    ;; IF TERMINATOR POP IT OFF THE STACK
3853 020674 012600      60$: MOV  (SP)+, RO  ;; RESTORE RO
3854 020676 062716 000002  3$: ADD  #2, (SP)  ;; ADJUST RETURN PC
3855 020702 000002      RTI      ;; RETURN
3856 020704 122716 000011  4$: CMPB #HT, (SP)  ;; BRANCH IF <HT>
3857 020710 001430      BEQ     8$      ;;
3858 020712 122716 000200  CMPB  #CRLF, (SP)  ;; BRANCH IF NOT <CRLF>
3859 020716 001006      BNE     5$      ;;
3860 020720 005726      TST    (SP)+    ;; POP <CR><LF> EQUIV
3861 020722 104401      TYPE    ;; TYPE A CR AND LF
3862 020724 001207      $CRLF
3863 020726 105037 021062  CLRB   $CHARCNT  ;; CLEAR CHARACTER COUNT
3864 020732 000755      BR      2$      ;; GET NEXT CHARACTER
3865 020734 004737 021016  5$: JSR   PC, $TYPEC  ;; GO TYPE THIS CHARACTER
3866 020740 123726 001156  6$: CMPB $FILLC, (SP)+  ;; IS IT TIME FOR FILLER CHARS.?
3867 020744 001350      BNE     2$      ;; IF NO GO GET NEXT CHAR.
3868 020746 013746 001154  MOV    $NULL, -(SP)  ;; GET # OF FILLER CHARS. NEEDED
3869                                AND THE NULL CHAR.
3870 020752 105366 000001  7$: DECB 1(SP)    ;; DOES A NULL NEED TO BE TYPED?
3871 020756 002770      BLT    6$      ;; BR IF NO--GO POP THE NULL OFF OF STACK
3872 020760 004737 021016  JSR   PC, $TYPEC  ;; GO TYPE A NULL
3873 020764 105337 021062  DECB  $CHARCNT  ;; DO NOT COUNT AS A COUNT
3874 020770 000770      BR      7$      ;; LOOP
3875
3876                                ;HORIZONTAL TAB PROCESSOR
3877
3878 020772 112716 000040  8$: MOVB #' (SP)    ;; REPLACE TAB WITH SPACE
3879 020776 004737 021016  9$: JSR   PC, $TYPEC  ;; TYPE A SPACE
3880 021002 132737 000007 021062  BITB  #7, $CHARCNT  ;; BRANCH IF NOT AT
3881 021010 001372      BNE     9$      ;; TAB STOP
3882 021012 005726      TST    (SP)+    ;; POP SPACE OFF STACK
3883 021014 000724      BR      2$      ;; GET NEXT CHARACTER
3884 021016 105777 160126  $TYPEC: TSTB 2$TPS  ;; WAIT UNTIL PRINTER IS READY
3885 021022 100375      BPL    $TYPEC
3886 021024 116677 000002 160120  MOVB  2(SP), 2$TPB  ;; LOAD CHAR TO BE TYPED INTO DATA REG.
3887 021032 122766 000015 000002  CMPB  #CR, 2(SP)  ;; IS CHARACTER A CARRIAGE RETURN?
3888 021040 001003      BNE     1$      ;; BRANCH IF NO
3889 021042 105037 021062  CLRB  $CHARCNT  ;; YES--CLEAR CHARACTER COUNT
3890 021046 000406      BR      $TYPEX  ;; EXIT
3891 021050 122766 000012 000002 1$: CMPB #LF, 2(SP)  ;; IS CHARACTER A LINE FEED?
3892 021056 001402      BEQ    $TYPEX  ;; BRANCH IF YES
3893 021060 105227      INCB  (PC)+    ;; COUNT THE CHARACTER
3894 021062 000000  $CHARCNT: .WORD 0  ;; CHARACTER COUNT STORAGE
3895 021064 000207  $TYPEX: RTS    PC
3896
3897                                .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
3898
3899                                ;*****
3900                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
    
```

# H06

```

3901 ;#OCTAL (ASCII) NUMBER AND TYPE IT.
3902 ;$STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3903 ;#CALL:
3904 ;#     MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
3905 ;#     TYPOS   ;;CALL FOR TYPEOUT
3906 ;#     .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3907 ;#     .BYT   M              ;;M=1 OR 0
3908 ;#                                     ;;1=TYPE LEADING ZEROS
3909 ;#                                     ;;0=SUPPRESS LEADING ZEROS
3910 ;#
3911 ;$STYPOS---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3912 ;$STYPOS OR $STYPOC
3913 ;#CALL:
3914 ;#     MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
3915 ;#     TYPON   ;;CALL FOR TYPEOUT
3916 ;#
3917 ;$STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3918 ;#CALL:
3919 ;#     MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
3920 ;#     TYPOC   ;;CALL FOR TYPEOUT
3921 ;#
3922 021066 017646 000000          $STYPOS: MOV     2(SP),-(SP)      ;;PICKUP THE MODE
3923 021072 116637 000001 021311 MOVVB  1(SP),%SOFILL    ;;LOAD ZERO FILL SWITCH
3924 021100 112637 021313          MOVVB  (SP)+,%SOMODE+1  ;;NUMBER OF DIGITS TO TYPE
3925 021104 062716 000002          ADD     #2,(SP)      ;;ADJUST RETURN ADDRESS
3926 021110 000406          BR     $TYPON
3927 021112 112737 000001 021311 $STYPOC: MOVVB  #1,%SOFILL    ;;SET THE ZERO FILL SWITCH
3928 021120 112737 000006 021313 MOVVB  #6,%SOMODE+1  ;;SET FOR SIX(6) DIGITS
3929 021126 112737 000005 021310 $STYPON: MOVVB  #5,%SOCNT  ;;SET THE ITERATION COUNT
3930 021134 010346          MOV     R3,-(SP)      ;;SAVE R3
3931 021136 010446          MOV     R4,-(SP)      ;;SAVE R4
3932 021140 010546          MOV     R5,-(SP)      ;;SAVE R5
3933 021142 113704 021313          MOVVB  %SOMODE+1,R4  ;;GET THE NUMBER OF DIGITS TO TYPE
3934 021146 005404          NEG     R4
3935 021150 062704 000006          ADD     #6,R4        ;;SUBTRACT IT FOR MAX. ALLOWED
3936 021154 110437 021312          MOVVB  R4,%SOMODE    ;;SAVE IT FOR USE
3937 021160 113704 021311          MOVVB  %SOFILL,R4    ;;GET THE ZERO FILL SWITCH
3938 021164 016605 000012          MOV     12(SP),R5    ;;PICKUP THE INPUT NUMBER
3939 021170 005003          CLR     R3           ;;CLEAR THE OUTPUT WORD
3940 021172 006105          15:   ROL     R5      ;;ROTATE MSB INTO "C"
3941 021174 000404          BR     3$           ;;GO DO MSB
3942 021176 006105          25:   ROL     R5      ;;FORM THIS DIGIT
3943 021200 006105          ROL     R5
3944 021202 006105          ROL     R5
3945 021204 010503          MOV     R5,R3
3946 021206 006103          35:   ROL     R3      ;;GET LSB OF THIS DIGIT
3947 021210 105337 021312          DECB   %SOMODE      ;;TYPE THIS DIGIT?
3948 021214 100016          BPL    7$           ;;BR IF NO
3949 021216 042703 177770          BIC    #177770,R3   ;;GET RID OF JUNK
3950 021222 001002          BNE    4$           ;;TEST FOR 0
3951 021224 005704          TST    R4           ;;SUPPRESS THIS 0?
3952 021226 001403          BEQ    5$           ;;BR IF YES
3953 021230 005204          45:   INC     R4      ;;DON'T SUPPRESS ANYMORE 0'S
3954 021232 052703 000060          BIS    #'0,R3      ;;MAKE THIS DIGIT ASCII
3955 021236 052703 000040          55:   BIS    #' ,R3  ;;MAKE ASCII IF NOT ALREADY
3956 021242 110337 021306          MOVVB  R3,%S        ;;SAVE FOR TYPING
  
```

```

3957 021246 104401 021306
3958 021252 105337 021310
3959 021256 003347
3960 021260 002402
3961 021262 005204
3962 021264 000744
3963 021266 012605
3964 021270 012604
3965 021272 012603
3966 021274 016666 000002 000004
3967 021302 012616
3968 021304 000002
3969 021306 000
3970 021307 000
3971 021310 000
3972 021311 000
3973 021312 000000
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986 021314
3987 021314 010046
3988 021316 010146
3989 021320 010246
3990 021322 010346
3991 021324 010546
3992 021326 012746 020200
3993 021332 016605 000020
3994 021336 100004
3995 021340 005405
3996 021342 112766 000055 000001
3997 021350 005000
3998 021352 012703 021530
3999 021356 112723 000040
4000 021362 005002
4001 021364 016001 021520
4002 021370 160105
4003 021372 002402
4004 021374 005202
4005 021376 000774
4006 021400 060105
4007 021402 005702
4008 021404 001002
4009 021406 105716
4010 021410 100407
4011 021412 106316
4012 021414 103003

75: TYPE BS
      DECB $OCNT
      BGT 25
      BLT 65
      INC R4
      BR 25
65: MOV (SP)+,R5
      MOV (SP)+,R4
      MOV (SP)+,R3
      MOV 2(SP),4(SP)
      MOV (SP)+,(SP)
      RTI
BS: .BYTE 0
     .BYTE 0
$OCNT: .BYTE 0
$OFILL: .BYTE 0
$OMODE: .WORD 0
.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

$TYPDS: MOV R0,-(SP) ;;PUSH R0 ON STACK
        MOV R1,-(SP) ;;PUSH R1 ON STACK
        MOV R2,-(SP) ;;PUSH R2 ON STACK
        MOV R3,-(SP) ;;PUSH R3 ON STACK
        MOV R5,-(SP) ;;PUSH R5 ON STACK
        MOV #20200,-(SP) ;;SET BLANK SWITCH AND SIGN
        MOV 20(SP),R5 ;;GET THE INPUT NUMBER
        BPL 15 ;;BR IF INPUT IS POS.
        NEG R5 ;;MAKE THE BINARY NUMBER POS.
        MOVB #'-',1(SP) ;;MAKE THE ASCII NUMBER NEG.
15: CLR R0 ;;ZERO THE CONSTANTS INDEX
     MOV #506LK,R3 ;;SETUP THE OUTPUT POINTER
     MOVB #'',(R3)+ ;;SET THE FIRST CHARACTER TO A BLANK
25: CLR R2 ;;CLEAR THE BCD NUMBER
     MOV $DTBL(R0),R1 ;;GET THE CONSTANT
35: SUB R1,R5 ;;FORM THIS BCD DIGIT
     BLT 45 ;;BR IF DONE
     INC R2 ;;INCREASE THE BCD DIGIT BY 1
45: ADD R1,R5 ;;ADD BACK THE CONSTANT
     TST R2 ;;CHECK IF BCD DIGIT=0
     BNE 55 ;;FALL THROUGH IF 0
     TSTB (SP) ;;STILL DOING LEADING 0'S?
4010 BMI 75 ;;BR IF YES
55: ASLB (SP) ;;MSD?
     BCC 65 ;;BR IF NO

```

# JOB

MD-11-DZRJF-A, RPO4/5/6 DUAL CONTRLR LOGIC TEST - PART 2  
 DZRJFA.CMB 02-NOV-76 19:11

MACY11 27(1006) 02-NOV-76 19:12 PAGE 74

CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

4013 021416 116663 000001 177777      MOVB    1(SP),-1(R3)      ;;YES--SET THE SIGN
4014 021424 052702 000060          6$:    BIS    #'0,R2      ;;MAKE THE BCD DIGIT ASCII
4015 021430 052702 000040          7$:    BIS    #' ,R2      ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
4016 021434 110223          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
4017 021436 005720          TST    (R0)+          ;;JUST INCREMENTING
4018 021440 020027 000010          CMP    R0,#10        ;;CHECK THE TABLE INDEX
4019 021444 002746          BLT    2$            ;;GO DO THE NEXT DIGIT
4020 021446 003052          BGT    8$            ;;GO TO EXIT
4021 021450 010502          MOV    R5,R2         ;;GET THE LSD
4022 021452 000764          BR     6$            ;;GO CHANGE TO ASCII
4023 021454 105726          8$:    TSTB   (SP)+      ;;WAS THE LSD THE FIRST NON-ZERO?
4024 021456 100003          BPL    9$            ;;BR IF NO
4025 021460 116663 177777 177776      MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
4026 021466 105013          9$:    CLRB   (R3)      ;;SET THE TERMINATOR
4027 021470 012605          MOV    (SP)+,R5      ;;POP STACK INTO R5
4028 021472 012603          MOV    (SP)+,R3      ;;POP STACK INTO R3
4029 021474 012602          MOV    (SP)+,R2      ;;POP STACK INTO R2
4030 021476 012601          MOV    (SP)+,R1      ;;POP STACK INTO R1
4031 021500 012600          MOV    (SP)+,R0      ;;POP STACK INTO R0
4032 021502 104401 021530          TYPE   $DBLK         ;;NOW TYPE THE NUMBER
4033 021506 016666 000002 000004      MOV    2(SP),4(SP)    ;;ADJUST THE STACK
4034 021514 012616          MOV    (SP)+,(SP)
4035 021516 000002          RTI
4036 021520 023420          $OTBL: 10000.        ;;RETURN TO USER
4037 021522 001750          1000.
4038 021524 000144          100.
4039 021526 000012          10.
4040 021530 000004          $DBLK: .BLKW 4
4041                                     .SBTTL TTY INPUT ROUTINE
4042
4043                                     ;*****
4044                                     .ENABL LSB
4045 021540 000000          $TKCNT: .WORD 0      ;;NUMBER OF ITEMS IN QUEUE
4046 021542 000000          $TKQIN: .WORD 0     ;;INPUT POINTER
4047 021544 000000          $TKQOUT: .WORD 0    ;;OUTPUT POINTER
4048 021546 000001          $TKQSRT: .BLKB 1    ;;TTY KEYBOARD QUEUE
4049                                     $TKQEND=.
4050                                     .EVEN
4051
4052                                     ;*TK INITIALIZE ROUTINE
4053                                     ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
4054                                     ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
4055
4056                                     ;*CALL:
4057                                     ;*      JSR    PC,$TKINT
4058                                     ;*      RETURN
4059
4060 021550 005037 021540          $TKINT: CLR    $TKCNT   ;;CLEAR COUNT OF ITEMS IN QUEUE
4061 021554 012737 021546 021542      MOV    # $TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
4062 021562 013737 021542 021544      MOV    $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
4063 021570 012737 021620 000060      MOV    # $TKSRT,$TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
4064 021576 012737 000200 000062      MOV    #200,$TKVEC+2  ;;BR" LEVEL 4
4065 021604 005777 157336          TST    $TKB          ;;CLEAR DONE FLAG
4066 021610 012777 000100 157326      MOV    #100,$TKS     ;;ENABLE TTY KEYBOARD INTERRUPT
4067 021616 000207          RTS    PC            ;;RETURN TO CALLER
4068

```



```

4069      ;*TK SERVICE ROUTINE
4070      ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
4071      ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
4072      ;*IT IN THE QUEUE.
4073
4074      021620 117746 157322 $TKSKV: MOVB  @STKB, -(SP)      ;: PICKUP THE CHARACTER
4075      021624 042716 177600      BIC  #1C177, (SP)      ;: STRIP THE JUNK
4076      021630 021627 000007      1S:  CMP  (SP), #7      ;: IS IT A CONTROL G?
4077      021634 001004      BNE  2S      ;: BRANCH IF NO
4078      021636 022737 000176 001140      CMP  #SWREG, SWR      ;: IS SOFT-SWR SELECTED?
4079      021644 001500      BEQ  6S      ;: GO TO SWR CHANGE
4080
4081      021646      2S:
4082      021646 022737 000001 021540      CMP  #1, STKCNT      ;: IS THE QUEUE FULL?
4083      021654 001004      BNE  3S      ;: BRANCH IF NO
4084      021656 104401 001202      TYPE $BELL      ;: RING THE TTY BELL
4085      021662 005726      TST  (SP)+      ;: CLEAN CHARACTER OFF OF STACK
4086      021664 000451      BR   5S      ;: EXIT
4087      021666 021627 000023      3S:  CMP  (SP), #23      ;: IS IT A CONTROL-S?
4088      021672 001021      BNE  32S      ;: BRANCH IF NO
4089      021674 005077 157244      CLR  @STKS      ;: DISABLE TTY KEYBOARD INTERRUPTS
4090      021700 005726      TST  (SP)+      ;: CLEAN CHAR OFF STACK
4091      021702 105777 157236      31S: TSTB @STKS      ;: WAIT FOR A CHAR
4092      021706 100375      BPL  31S      ;: LOOP UNTIL ITS THERE
4093      021710 117746 157232      MOVB @STKB, -(SP)      ;: GET THE CHARACTER
4094      021714 042716 177600      BIC  #1C177, (SP)      ;: MAKE IT 7-BIT ASCII
4095      021720 022627 000021      CMP  (SP)+, #21      ;: IS IT A CONTROL-Q?
4096      021724 001366      BNE  31S      ;: BRANCH IF NO
4097      021726 012777 000100 157210      MOV  #100, @STKS      ;: REENABLE TTY KEYBOARD INTERRUPTS
4098      021734 000002      RTI      ;: RETURN
4099      021736 005237 021540      32S: INC  STKCNT      ;: COUNT THIS CHARACTER
4100      021742 021627 000140      CMP  (SP), #140      ;: IS IT UPPER CASE?
4101      021746 002405      BLT  4S      ;: BRANCH IF YES
4102      021750 021627 000175      CMP  (SP), #175      ;: IS IT A SPECIAL CHAR?
4103      021754 003002      BGT  4S      ;: BRANCH IF YES
4104      021756 042716 000040      BIC  #40, (SP)      ;: MAKE IT UPPER CASE
4105      021762 112677 177554      4S:  MOVB (SP)+, @STKQIN      ;: AND PUT IT IN QUEUE
4106      021766 005237 021542      INC  STKQIN      ;: UPDATE THE POINTER
4107      021772 023727 021542 021547      CMP  STKQIN, #STKQEND ;: GO OFF THE END?
4108      022000 001003      BNE  5S      ;: BRANCH IF NO
4109      022002 012737 021546 021542      MOV  #STKQSR, STKQIN ;: RESET THE POINTER
4110      022010 000002      5S:  RTI      ;: RETURN
4111
4112      ;:*****
4113      ;: *SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
4114      ;: *ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
4115      ;: *SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
4116      ;: *CALL WHEN OPERATING IN TTY INTERRUPT MODE.
4117      022012 022737 000176 001140 $CKSWR: CMP  #SWREG, SWR      ;: IS THE SOFT-SWR SELECTED
4118      022020 001104      BNE  15S      ;: EXIT IF NOT
4119      022022 105777 157116      TSTB @STKS      ;: IS A CHAR WAITING?
4120      022026 100101      BPL  15S      ;: IF NOT, EXIT
4121      022030 117746 157112      MOVB @STKB, -(SP)      ;: YES
4122      022034 042716 177600      BIC  #1C177, (SP)      ;: MAKE IT 7-BIT ASCII
4123      022040 021627 000007      CMP  (SP), #7      ;: IS IT A CONTROL-G?
4124      022044 001300      BNE  2S      ;: IF NOT, PUT IT IN THE TTY QUEUE

```

```

4125                                     ;;AND EXIT
4126
4127                                     ;*****
4128                                     ;#CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
4129                                     ;#ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
4130                                     ;#CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
4131 022046 123727 001134 000001 6$:  CMPB  $AUTOB,#1      ;;ARE WE RUNNING IN AUTO-MODE?
4132 022054 001674          BEQ    2$              ;;BRANCH IF YES
4133 022056 005726          TST   (SP)+          ;;CLEAR CONTROL-G OFF STACK
4134 022060 004737 021550  JSR   PC,$TKINT      ;;FLUSH THE TTY INPUT QUEUE
4135 022064 005077 157054  CLR   2$TKS          ;;DISABLE TTY KEYBOARD INTERRUPTS
4136 022070 112737 000001 001135  MOVB  #1,$INTAG      ;;SET INTERRUPT MODE INDICATOR
4137
4138 022076 104401 022654          TYPE  , $CNTLG      ;;ECHO THE CONTROL-G (↑G)
4139 022102 104401 022661  $GTSWR: TYPE  , $MSWR      ;;TYPE CURRENT CONTENTS
4140 022106 013746 000176  MOV   $WREG,-(SP)   ;;SAVE SWREG FOR TYPEOUT
4141 022112 104402          TPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4142 022114 104401 022672  TYPE  , $MNEW      ;;PROMPT FOR NEW SWR
4143 022120 005046          19$: CLR  -(SP)        ;;CLEAR COUNTER
4144 022122 005046          CLR  -(SP)        ;;THE NEW SWR
4145 022124 105777 157014  7$:  TSTB 2$TKS      ;;CHAR THERE?
4146 022130 100375          BPL   7$          ;;IF NOT TRY AGAIN
4147
4148 022132 117746 157010  MOVB  2$TKB,-(SP)   ;;PICK UP CHAR
4149 022136 042716 177600  BIC   #1C177,(SP)  ;;MAKE IT 7-BIT ASCII
4150
4151
4152
4153 022142 021627 000025          9$:  CMP   (SP),#25   ;;IS IT A CONTROL-U?
4154 022146 001005          BNE   10$          ;;BRANCH IF NOT
4155 022150 104401 022647          TYPE  , $CNTLU    ;;YES, ECHO CONTROL-U (↑U)
4156 022154 062706 000006  20$: ADD   #6,SP     ;;IGNORE PREVIOUS INPUT
4157 022160 000757          BR    19$         ;;LET'S TRY IT AGAIN
4158
4159
4160 022162 021627 000015          10$: CMP   (SP),#15   ;;IS IT A <CR>?
4161 022166 001022          BNE   16$          ;;BRANCH IF NO
4162 022170 005766 000004          TST   4(SP)        ;;YES, IS IT THE FIRST CHAR?
4163 022174 001403          BEQ   11$          ;;BRANCH IF YES
4164 022176 016677 000002 156734  MOV   2(SP),2$SWR   ;;SAVE NEW SWR
4165 022204 062706 000006          11$: ADD   #6,SP     ;;CLEAR UP STACK
4166 022210 104401 001207          14$: TYPE  , $CRLF   ;;ECHO <CR> AND <LF>
4167 022214 123727 001135 000001  CMPB  $INTAG,#1    ;;RE-ENABLE TTY KBD INTERRUPTS?
4168 022222 001003          BNE   15$          ;;BRANCH IF NOT
4169 022224 012777 000100 156712  MOV   #100,2$TKS   ;;RE-ENABLE TTY KBD INTERRUPTS
4170 022232 000002          RTI          ;;RETURN
4171 022234 004737 021016          15$: JSR   PC,$TYPEC   ;;ECHO CHAR
4172 022240 021627 000060          16$: CMP   (SP),#60   ;;CHAR < 0?
4173 022244 002420          BLT   18$          ;;BRANCH IF YES
4174 022246 021627 000067          CMP   (SP),#67    ;;CHAR > 7?
4175 022252 003015          BGT   18$          ;;BRANCH IF YES
4176 022254 042726 000060          BIC   #60,(SP)+   ;;STRIP-OFF ASCII
4177 022260 005766 000002          TST   2(SP)       ;;IS THIS THE FIRST CHAR
4178 022264 001403          BEQ   17$          ;;BRANCH IF YES
4179 022266 006316          ASL   (SP)        ;;NO, SHIFT PRESENT
4180 022270 006316          ASL   (SP)        ;;CHAR OVER TO MAKE

```

```

4191 022272 006316          ASL      (SP)          ;; ROOM FOR NEW ONE.
4192 022274 005266 000002 17$: INC      2(SP)          ;; KEEP COUNT OF CHAR
4193 022300 056616 177776   BIS      -2(SP), (SP)  ;; SET IN NEW CHAR
4194 022304 000707          BR       7$           ;; GET THE NEXT ONE
4195 022306 104401 001206   18$: TYPE  $QUES      ;; TYPE ?(CR)(LF)
4196 022312 000720          BR       20$         ;; SIMULATE CONTROL-U
4197 .DSABL  LSB

;*****
;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
;CALL:
;   RDCHR          ;; GET A CHARACTER FROM THE QUEUE
;   RETURN HERE   ;; CHARACTER IS ON THE STACK
;                 ;; WITH PARITY BIT STRIPPED OFF
;
$RDCHR: MOV      (SP), -(SP)  ;; PUSH DOWN THE PC AND
        MOV      4(SP), 2(SP) ;; THE PS
        CLR      4(SP)       ;; GET READY FOR A CHARACTER
        CLR      -(SP)      ;; PUT NEW PS ON STACK
        MOV      #64$, -(SP) ;; PUT NEW PC ON STACK
        RTI          ;; POP NEW PC AND PS
64$:
1$: TST      $STKCNT        ;; WAIT ON A CHARACTER
    BEQ      1$
    DEC      $STKCNT        ;; DECREMENT THE COUNTER
    MOV      2$STKQOUT, 4(SP) ;; GET ONE CHARACTER
    INC      $STKQOUT       ;; UPDATE THE POINTER
    CMP      $STKQOUT, #STKQEND ;; DID IT GO OFF OF THE END?
    BNE      2$            ;; BRANCH IF NO
    MOV      #STKQSR, $STKQOUT ;; RESET THE POINTER
    RTI          ;; RETURN
2$:
;*****
;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
;CALL:
;   RDLIN         ;; INPUT A STRING FROM THE TTY
;   RETURN HERE   ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
;                 ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
;
$RDLIN: MOV      R3, -(SP)   ;; SAVE R3
        CLR      -(SP)     ;; CLEAR THE RUBOUT KEY
1$: MOV      #STTYIN, R3    ;; GET ADDRESS
2$: CMP      #STTYIN+7, R3  ;; BUFFER FULL?
        BLOS    4$         ;; BR IF YES
        RDCHR    ;; GO READ ONE CHARACTER FROM THE TTY
        MOV      (SP)+, (R3) ;; GET CHARACTER
10$: CMP      #177, (R3)    ;; IS IT A RUBOUT
        BNE    5$         ;; BR IF NO
        TST     (SP)       ;; IS THIS THE FIRST RUBOUT?
        BNE    6$         ;; BR IF NO
        MOV     #' \, 9$   ;; TYPE A BACK SLASH
        MOV     #-1, (SP)  ;; SET THE RUBOUT KEY
6$: DEC      R3            ;; BACKUP BY ONE
        CMP     R3, #STTYIN ;; STACK EMPTY?
    
```

```

4237 022464 103434          BLO      4$          ;; BR IF YES
4238 022466 111337 022636  MOVB    (R3),9$     ;; SETUP TO TYPEOUT THE DELETED CHAR.
4239 022472 104401 022636  TYPE    9$          ;; GO TYPE
4240 022476 000746          BR      2$          ;; GO READ ANOTHER CHAR.
4241 022500 005716          5$: TST    (SP)        ;; RUBOUT KEY SET?
4242 022502 001406          BEQ    7$          ;; BR IF NO
4243 022504 112737 000134 022636  MOVB    '\,9$      ;; TYPE A BACK SLASH
4244 022512 104401 022636  TYPE    9$          ;;
4245 022516 005016          CLR    (SP)        ;; CLEAR THE RUBOUT KEY
4246 022520 122713 000025  7$: CMPB  #25,(R3)   ;; IS CHARACTER A CTRL U?
4247 022524 001003          BNE    8$          ;; BR IF NO
4248 022526 104401 022647  TYPE    SCNTLU     ;; TYPE A CONTROL "U"
4249 022532 000726          BR      1$          ;; GO START OVER
4250 022534 122713 000022  8$: CMPB  #22,(R3)   ;; IS CHARACTER A "R"?
4251 022540 001011          BNE    3$          ;; BRANCH IF NO
4252 022542 105013          CLRB   (R3)        ;; CLEAR THE CHARACTER
4253 022544 104401 001207  TYPE    ,SCRLF     ;; TYPE A "CR" & "LF"
4254 022550 104401 022640  TYPE    $TTYIN     ;; TYPE THE INPUT STRING
4255 022554 000717          BR      2$          ;; GO PICKUP ANOTHER CHARACTER
4256 022556 104401 001206  4$: TYPE    $QUES   ;; TYPE A '?'
4257 022562 000712          BR      1$          ;; CLEAR THE BUFFER AND LOOP
4258 022564 111337 022636  3$: MOVB   (R3),9$   ;; ECHO THE CHARACTER
4259 022570 104401 022636  TYPE    9$          ;;
4260 022574 122723 000015  CMPB   #15,(R3)+  ;; CHECK FOR RETURN
4261 022600 001305          BNE    2$          ;; LOOP IF NOT RETURN
4262 022602 105063 177777  CLRB   -1(R3)     ;; CLEAR RETURN (THE 15)
4263 022606 104401 001210  TYPE    $LF        ;; TYPE A LINE FEED
4264 022612 005726          TST   (SP)+       ;; CLEAN RUBOUT KEY FROM THE STACK
4265 022614 012603          MOV   (SP)+,R3    ;; RESTORE R3
4266 022616 011646          MOV   (SP)-,(SP)  ;; ADJUST THE STACK AND PUT ADDRESS OF THE
4267 022620 016666 000004 000002  MOV   4(SP),2(SP)  ;; FIRST ASCII CHARACTER ON IT
4268 022626 012766 022640 000004  MOV   #TTYIN,4(SP)
4269 022634 000002          RTI           ;; RETURN
4270 022636 000          9$: .BYTE  0       ;; STORAGE FOR ASCII CHAR. TO TYPE
4271 022637 000          .BYTE  0       ;; TERMINATOR
4272 022640 000007          $TTYIN: .BLKB  7  ;; RESERVE 7 BYTES FOR TTY INPUT
4273 022647 136 006525 000012  $CNTLU: .ASCIZ  /TU<(15)<(12)  ;; CONTROL "U"
4274 022654 043536 005015 000012  $CNTLG: .ASCIZ  /TG<(15)<(12)  ;; CONTROL "G"
4275 022661 015 051412 051127  $MSWR:  .ASCIZ  <(15)<(12)/SWR = /
4276 022666 036440 000040          $MNEW:  .ASCIZ  / NEW = /
4277 022672 020040 042516 020127
4278 022700 020075 000
4279 022704
4280 .EVEN
4281 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
4282
4283 ;; *****
4284 ;; *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
4285 ;; *CHANGE IT TO BINARY.
4286 ;; *THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
4287 ;; *OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
4288 ;; *FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
4289 ;; *THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
4290 ;; *CALL:
4291 ;; * RDOCT          ;; READ AN OCTAL NUMBER
4292 ;; * RETURN HERE   ;; LOW ORDER BITS ARE ON TOP OF THE STACK
;; *              ;; HIGH ORDER BITS ARE IN $HI OCT

```

```

4293
4294 022704 011646
4295 022706 016666 000004 000002
4296 022714 010046
4297 022716 010146
4298 022720 010246
4299 022722 104411
4300 022724 012600
4301 022726 010037 023032
4302 022732 005001
4303 022734 005002
4304 022736 112046
4305 022740 001420
4306 022742 122716 000060
4307 022746 003026
4308 022750 122716 000067
4309 022754 002423
4310 022756 006301
4311 022760 006102
4312 022762 006301
4313 022764 006102
4314 022766 006301
4315 022770 006102
4316 022772 042716 177770
4317 022776 062601
4318 023000 000756
4319 023002 005726
4320 023004 010166 000012
4321 023010 010237 023042
4322 023014 012602
4323 023016 012601
4324 023020 012600
4325 023022 000002
4326 023024 005726
4327 023026 105010
4328 023030 104401
4329 023032 000000
4330 023034 104401 001206
4331 023040 000730
4332 023042 000000
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348

$RDOCT: MOV (SP),-(SP) ;; PROVIDE SPACE FOR THE
MOV 4(SP),2(SP) ;; INPUT NUMBER
MOV RO,-(SP) ;; PUSH RO ON STACK
MOV R1,-(SP) ;; PUSH R1 ON STACK
MOV R2,-(SP) ;; PUSH R2 ON STACK
1$: RDLIN ;; READ AN ASCII LINE
MOV (SP)+,RO ;; GET ADDRESS OF 1ST CHARACTER
MOV RO,$$ ;; AND SAVE IT
CLR R1 ;; CLEAR DATA WORD
CLR R2
2$: MOVB (RO)+,-(SP) ;; PICKUP THIS CHARACTER
BEQ $$ ;; IF ZERO GET OUT
CMPB #'0,(SP) ;; MAKE SURE THIS CHARACTER
BGT $$ ;; IS AN OCTAL DIGIT
CMPB #'7,(SP)
BLT $$
ASL R1 ;; *2
ROL R2
ASL R1 ;; *4
ROL R2
ASL R1 ;; *8
ROL R2
BIC #'C7,(SP) ;; STRIP THE ASCII JUNK
ADD (SP)+,R1 ;; ADD IN THIS DIGIT
BR 2$ ;; LOOP
3$: TST (SP)+ ;; CLEAN TERMINATOR FROM STACK
MOV R1,12(SP) ;; SAVE THE RESULT
MOV R2,$HI OCT
MOV (SP)+,R2 ;; POP STACK INTO R2
MOV (SP)+,R1 ;; POP STACK INTO R1
MOV (SP)+,RO ;; POP STACK INTO RO
RTI ;; RETURN
4$: TST (SP)+ ;; CLEAN PARTIAL FROM STACK
CLRB (RO) ;; SET A TERMINATOR
TYPE ;; TYPE UP THRU THE BAD CHAR.
5$: .WORD 0
TYPE $QUES ;; "?" "CR" & "LF"
BR 1$ ;; TRY AGAIN
$HI OCT: .WORD 0 ;; HIGH ORDER BITS GO HERE
.SBTL SAVE AND RESTORE RO-R5 ROUTINES

*****
*SAVE RO-R5
*CALL:
* SAVREG
*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
*
*TOP---(+16)
* +2---(+18)
* +4---R5
* +6---R4
* +8---R3
*+10---R2
*+12---R1
*+14---RO

```

SAVE AND RESTORE RO-R5 ROUTINES

4349  
 4350 023044  
 4351 023044 010046  
 4352 023046 010146  
 4353 023050 010246  
 4354 023052 010346  
 4355 023054 010446  
 4356 023056 010546  
 4357 023060 016646 000022  
 4358 023064 016646 000022  
 4359 023070 016646 000022  
 4360 023074 016646 000022  
 4361 023100 000002

```

$SAVREG:
MOV RO,-(SP)      ;; PUSH RO ON STACK
MOV R1,-(SP)      ;; PUSH R1 ON STACK
MOV R2,-(SP)      ;; PUSH R2 ON STACK
MOV R3,-(SP)      ;; PUSH R3 ON STACK
MOV R4,-(SP)      ;; PUSH R4 ON STACK
MOV R5,-(SP)      ;; PUSH R5 ON STACK
MOV 22(SP),-(SP)  ;; SAVE PS OF MAIN FLOW
MOV 22(SP),-(SP)  ;; SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP)  ;; SAVE PS OF CALL
MOV 22(SP),-(SP)  ;; SAVE PC OF CALL
RTI
  
```

4362  
 4363  
 4364  
 4365  
 4366 023102  
 4367 023102 012666 000022  
 4368 023106 012666 000022  
 4369 023112 012666 000022  
 4370 023116 012666 000022  
 4371 023122 012605  
 4372 023124 012604  
 4373 023126 012603  
 4374 023130 012602  
 4375 023132 012601  
 4376 023134 012600  
 4377 023136 000002

```

; *RESTORE RO-R5
; *CALL:
; * RESREG
$RESREG:
MOV (SP)+,22(SP)  ;; RESTORE PC OF CALL
MOV (SP)+,22(SP)  ;; RESTORE PS OF CALL
MOV (SP)+,22(SP)  ;; RESTORE PC OF MAIN FLOW
MOV (SP)+,22(SP)  ;; RESTORE PS OF MAIN FLOW
MOV (SP)+,R5      ;; POP STACK INTO R5
MOV (SP)+,R4      ;; POP STACK INTO R4
MOV (SP)+,R3      ;; POP STACK INTO R3
MOV (SP)+,R2      ;; POP STACK INTO R2
MOV (SP)+,R1      ;; POP STACK INTO R1
MOV (SP)+,R0      ;; POP STACK INTO R0
RTI
  
```

4378  
 4379  
 4380  
 4381  
 4382  
 4383  
 4384  
 4385

```

.SBTTL TRAP DECODER

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

4386 023140 010046  
 4387 023142 016600 000002  
 4388 023146 005740  
 4389 023150 111000  
 4390 023152 006300  
 4391 023154 016000 023174  
 4392 023160 000200

```

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

4393  
 4394  
 4395  
 4396  
 4397 023162 011646  
 4398 023164 016666 000004 000002  
 4399 023172 000002

```

;; THIS IS USE TO HANDLE THE "GETPRI" MACRO
$TRAP2:
MOV (SP),-(SP)    ;; MOVE THE PC DOWN
MOV 4(SP),2(SP)   ;; MOVE THE PSW DOWN
RTI               ;; RESTORE THE PSW
  
```

4400  
 4401  
 4402  
 4403  
 4404

```

.SBTTL TRAP TABLE

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



LINE	ADDRESS	ROUTINE
005		
006		
007		
008	023174 023162	\$TRPAD: .WORD \$TRAP2
009	023176 020646	\$TYPE ::CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
010	023200 021112	\$TYPOC ::CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
011	023202 021066	\$TYPOS ::CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
012	023204 021126	\$TYPON ::CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
013	023206 021314	\$TYPDS ::CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
014		
015	023210 022102	\$GTSWR ;;CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
016		
017	023212 022012	\$CKSWR ::CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
018	023214 022314	\$RDCHR ::CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
019	023216 022404	\$ROLIN ::CALL=ROLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
020	023220 022704	\$RDOCT ::CALL=RDOCT TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
021	023222 023044	\$SAVREG ::CALL=SAVREG TRAP+13(104413) SAVE RO-RS ROUTINE
022	023224 023102	\$RESREG ::CALL=RESREG TRAP+14(104414) RESTORE RO-RS ROUTINE
023		
024		
025		
026		
027		
028		
029		
030	023226 005015 046412 044501	TITLE: .ASCII (CR)(LF)(LF)/MAINDEC-11-DZRJF-A/(CR)(LF)
031	023234 042116 041505 030455	
032	023242 026461 055104 045122	
033	023250 026506 006501 012	
034	023255 122 030120 027464	.ASCIIZ 0RPO4/5/6 DUAL CONTROLLER LOGIC TEST - PART 2(CR)(LF)(LF)
035	023262 027465 020066 052504	
036	023270 046101 041440 047117	
037	023276 051124 046117 042514	
038	023304 020122 047514 044507	
039	023312 020103 042524 052123	
040	023320 026440 050040 051101	
041	023326 020124 006462 005012	
042	023334 000	
043	023335 015 042412 052116	ENTERA: .ASCIIZ (CR)(LF)/ENTER DRIVE ADDRESS: /
044	023342 051105 042040 044522	
045	023350 042526 040440 042104	
046	023356 042522 051523 020072	
047	023364 000	
048	023365 111 053116 046101	ADRERR: .ASCIIZ /INVALID ADDRESS/(CR)(LF)
049	023372 042111 040440 042104	
050	023400 042522 051523 005015	
051	023406 000	
052	023407 015 050012 051117	PORTAIS: .ASCIIZ (CR)(LF)/PORT A ADDRESS IS: /
053	023414 020124 020101 042101	
054	023422 051104 051505 020123	
055	023430 051511 020072 000	
056	023435 015 050012 051117	PORTBIS: .ASCIIZ (CR)(LF)/PORT B ADDRESS IS: /
057	023442 020124 020102 042101	
058	023450 051104 051505 020123	
059	023456 051511 020072 000	
060	023463 015 051412 051531	NOCLOCK: .ASCIIZ (CR)(LF)/SYSTEM MUST HAVE 'L' OR 'P' CLOCK/(CR)(LF)(LF)

E07

MD-11-DZRJF-A, RPO4/5/6 DUAL CONTRLR LOGIC TEST - PART 2  
DZRJFA.CMB 02-NOV-76 19:11 TELETYPE MESSAGES

MACY11 27(1006) 02-NOV-76 19:12 PAGE 82

4461	023470	042524	020115	052515	
4462	023476	052123	044040	053101	
4463	023504	020105	046047	020047	
4464	023512	051117	023440	023520	
4465	023520	041440	047514	045503	
4466	023526	005015	000012		
4467	023533	042412	052116	051105	TESTNO: .ASCIZ <LF>/ENTER TEST #: /
4468	023540	052040	051505	020124	
4469	023546	035043	000040		
4470	023552	047111	040526	044514	BADNO: .ASCIZ /INVALID TEST NUMBER/<CR><LF>
4471	023560	020104	042522	052123	
4472	023566	047040	046525	042502	
4473	023574	006522	000012		
4474	023600	005015	052012	042510	ADDRIS: .ASCIZ <CR><LF><LF>/THE PRESENT ADDRESS OF THE RH11 (RPCS1) IS: /
4475	023606	050040	042522	042523	
4476	023614	052116	040440	042104	
4477	023622	042522	051523	047440	
4478	023630	020106	044124	020105	
4479	023636	044122	030461	024040	
4480	023644	050123	051503	024461	
4481	023652	044440	035123	000040	
4482	023660	042412	052116	051105	NTRH11: .ASCIZ <LF>/ENTER NEW RH11 ADDRESS: /
4483	023666	047040	053505	051040	
4484	023674	030510	020061	042101	
4485	023702	051104	051505	035123	
4486	023710	000040			
4487	023712	005015	050012	042522	STANDBY: .ASCII <CR><LF><LF>/PRESS 'STANDBY' ON DRIVE/
4488	023720	051523	023440	052123	
4489	023726	047101	041104	023531	
4490	023734	047440	020116	051104	
4491	023742	053111	105		
4492	023745	015	050012	047522	.ASCIZ <CR><LF>/PROGRAM WILL LOOP WAITING FOR 'MOL' TO SET FROM PORT /
4493	023752	051107	046501	053440	
4494	023760	046111	020114	047514	
4495	023766	050117	053440	044501	
4496	023774	044524	043516	043040	
4497	024002	051117	023440	047515	
4498	024010	023514	052040	020117	
4499	024016	042523	020124	051106	
4500	024024	046517	050040	051117	
4501	024032	020124	000		
4502	024038	015	055012	042522	SWTCHN: .ASCIZ <CR><LF><LF>/RETURN 'CONTROLLER SELECT' SWITCH ON DRIVE TO 'A/B'@
4503	024042	052524	047122	023440	
4504	024050	047503	052116	047522	
4505	024056	046114	051105	051440	
4506	024064	046105	041505	023524	
4507	024072	051440	044527	041524	
4508	024100	020110	047117	042040	
4509	024106	044522	042526	052040	
4510	024114	020117	040447	041057	
4511	024122	000047			
4512	024124	005015	052012	051125	SWTCHA: .ASCIZ <CR><LF><LF>/TURN 'CONTROLLER SELECT' SWITCH ON DRIVE TO 'A'/'
4513	024132	020116	041447	047117	
4514	024140	051124	046117	042514	
4515	024146	020122	042523	042514	
4516	024154	052103	020047	053523	

4517	024162	052111	044103	047440
4518	024170	020116	051104	053111
4519	024176	020105	047524	023440
4520	024204	023501	000	
4521	024207	015	005012	052524
4522	024214	047122	023440	047503
4523	024222	052116	047522	046114
4524	024230	051105	051440	046105
4525	024236	041505	023524	051440
4526	024244	044527	041524	020110
4527	024250	047117	042040	044522
4528	024260	042526	052040	020117
4529	024266	041047	000047	
4530	024272	050015	044124	047105
4531	024300	050040	042522	051523
4532	024306	023440	047503	052116
4533	024314	020047	047117	052040
4534	024322	042510	050040	047522
4535	024330	042503	051523	051117
4536	024336	005015	000	
4537	024341	015	005012	052123
4538	024346	050117	052040	042510
4539	024354	042040	044522	042526
4540	024362	000		
4541	024363	015	005012	052123
4542	024370	051101	020124	044124
4543	024376	020105	051104	053111
4544	024404	020105	020055	044124
4545	024412	020105	051120	043517
4546	024420	040522	020115	044527
4547	024426	046114	053440	044501
4548	024434	020124	047506	020122
4549	024442	046447	046117	020047
4550	024450	047524	051440	052105
4551	024456	000		
4552				
4553				
4554				
4555				
4556				
4557				
4558				
4559	024457	104	044522	042526
4560	024464	044440	020123	047516
4561	024472	026516	054105	051511
4562	024500	042524	052116	024040
4563	024506	047047	042105	020047
4564	024514	044502	020124	042523
4565	024522	024524	000	
4566				
4567	024525	127	047522	043516
4568	024532	042040	044522	042526
4569	024540	052040	050131	000105
4570				
4571	024546	047503	052116	047522
4572	024554	046114	051105	051440

SWTCHB: .ASCIZ <CR><LF><LF>/TURN 'CONTROLLER SELECT' SWITCH ON DRIVE TO 'B'/

CONTUE: .ASCIZ <CR><LF>/THEN PRESS 'CONT' ON THE PROCESSOR/<CR><LF>

CYCLED: .ASCIZ <CR><LF><LF>/STOP THE DRIVE/

CYCLEU: .ASCIZ <CR><LF><LF>/START THE DRIVE - THE PROGRAM WILL WAIT FOR 'MOL' TO SET/

::\*\*\*\*\*

.SBTTL TEST ERROR MESSAGES

::\*\*\*\*\*

EM1: .ASCIZ /DRIVE IS NON-EXISTENT ('MED' BIT SET)/

EM2: .ASCIZ /WRONG DRIVE TYPE/

EM3: .ASCIZ @CONTROLLER SELECT SWITCH ON DRIVE NOT IN 'A/B'@

4573	024562	046105	041505	020124		
4574	024570	053523	052111	044103		
4575	024576	047440	020116	051104		
4576	024604	053111	020105	047516		
4577	024612	020124	047111	023440		
4578	024620	027501	023502	000		
4579						
4580	024625	104	044522	042526	EM4:	.ASCIZ /DRIVE NOT ON LINE/
4581	024632	047040	052117	047440		
4582	024640	020116	044514	042516		
4583	024646	000				
4584						
4585	024647	123	051105	040511	EM5:	.ASCIZ /SERIAL NUMBER READ THROUGH EACH PORT NOT THE SAME/
4586	024654	020114	052516	041115		
4587	024662	051105	051040	040505		
4588	024670	020104	044124	047522		
4589	024676	043525	020110	040505		
4590	024704	044103	050040	051117		
4591	024712	020124	047516	020124		
4592	024720	044124	020105	040523		
4593	024726	042515	000			
4594						
4595	024731	124	046511	047505	EM6:	.ASCIZ /TIMEOUT HAS NOT OCCURRED WITHIN 2 SECONDS/
4596	024736	052125	044040	051501		
4597	024744	047040	052117	047440		
4598	024752	041503	051125	042522		
4599	024760	020104	044527	044124		
4600	024766	047111	031040	051440		
4601	024774	041505	047117	051504		
4602	025002	000				
4603						
4604	025003	124	046511	047505	EM7:	.ASCIZ /TIMEOUT ONE-SHOT IS LESS THAN 500 MS/
4605	025010	052125	047440	042516		
4606	025016	051455	047510	020124		
4607	025024	051511	046040	051505		
4608	025032	020123	044124	047101		
4609	025040	032440	030060	046440		
4610	025046	000123				
4611						
4612	025050	042522	042101	047111	EM10:	.ASCIZ /READIN PRESET DOES NOT SET VOLUME VALID FOR THE PORT/
4613	025056	050040	042522	042523		
4614	025064	020124	047504	051505		
4615	025072	047040	052117	051440		
4616	025100	052105	053040	046117		
4617	025106	046525	020105	040526		
4618	025114	044514	020104	047506		
4619	025122	020122	044124	020105		
4620	025130	047520	052122	000		
4621						
4622	025135	047	047507	020047	EM11:	.ASCIZ /'GO' BIT RESET DURING UNLOAD COMMAND/
4623	025142	044502	020124	042522		
4624	025150	042523	020124	052504		
4625	025156	044522	043516	052440		
4626	025164	046116	040517	020104		
4627	025172	047503	046515	047101		
4628	025200	000104				

4629					
4630	025202	047111	047503	051122	EM12: .ASCIZ /INCORRECT STATUS DURING UNLOAD COMMAND/
4631	025210	041505	020124	052123	
4632	025216	052101	051525	042040	
4633	025224	051125	047111	020107	
4634	025232	047125	047514	042101	
4635	025240	041440	046517	040515	
4636	025246	042116	000		
4637					
4638	025251	104	044522	042526	EM13: .ASCIZ /DRIVE DID NOT RETURN TO NEUTRAL AFTER UNLOAD COMMAND/
4639	025256	042040	042111	047040	
4640	025264	052117	051040	052105	
4641	025272	051125	020116	047524	
4642	025300	047040	052505	051124	
4643	025306	046101	040440	052106	
4644	025314	051105	052440	046116	
4645	025322	040517	020104	047503	
4646	025330	046515	047101	000104	
4647					
4648	025336	052101	042524	052116	EM14: .ASCIZ /ATTENTION BIT SET ON 'OPPOSITE PORT' AFTER UNLOAD/
4649	025344	047511	020116	044502	
4650	025352	020124	042523	020124	
4651	025360	047117	023440	050117	
4652	025366	047520	044523	042524	
4653	025374	050040	051117	023524	
4654	025402	040440	052106	051105	
4655	025410	052440	046116	040517	
4656	025416	000104			
4657					
4658	025420	052101	042524	052116	EM15: .ASCIZ /ATTENTION BIT NOT SET ON PORT WHICH ISSUED 'UNLOAD'/
4659	025426	047511	020116	044502	
4660	025434	020124	047516	020124	
4661	025442	042523	020124	047117	
4662	025450	050040	051117	020124	
4663	025456	044127	041511	020110	
4664	025464	051511	052523	042105	
4665	025472	023440	047125	047514	
4666	025500	042101	000047		
4667					
4668	025504	051104	053111	020105	EM16: .ASCII /DRIVE NOT IN NEUTRAL AFTER UNLOAD WITH 'CONTROLLER/<CR><LF>
4669	025512	047516	020124	047111	
4670	025520	047040	052505	051124	
4671	025526	046101	040440	052106	
4672	025534	051105	052440	046116	
4673	025542	040517	020104	044527	
4674	025550	044124	023440	047503	
4675	025556	052116	047522	046114	
4676	025564	051105	005015		
4677	025570	042523	042514	052103	.ASCIZ @SELECT' SWITCH MOVED FROM 'A/B'@
4678	025576	020047	053523	052111	
4679	025604	044103	046440	053117	
4680	025612	042105	043040	047522	
4681	025620	020115	040447	041057	
4682	025626	000047			
4683					
4684	025630	051104	053111	020105	EM17: .ASCIZ /DRIVE LOCKED ON PORT 'A' BY SWITCH WHILE CYCLED UP/

4685	025636	047514	045503	042105	
4686	025644	047440	020116	047520	
4687	025652	052122	023440	023501	
4688	025660	041040	020131	053523	
4689	025666	052111	044103	053440	
4690	025674	044510	042514	041440	
4691	025702	041531	042514	020104	
4692	025710	050125	000		
4693					
4694	025713	104	044522	042526	EM20: .ASCIZ /DRIVE LOCKED ON PORT 'B' BY SWITCH WHILE CYCLED UP/
4695	025720	046040	041517	042513	
4696	025726	020104	047117	050040	
4697	025734	051117	020124	041047	
4698	025742	020347	054502	051440	
4699	025750	044527	041524	020110	
4700	025756	044127	046111	020105	
4701	025764	054503	046103	042165	
4702	025772	052440	000120		
4703					
4704	025776	052123	052101	051525	EM21: .ASCIZ /STATUS INCORRECT FOR PORT AFTER CYCLE UP/
4705	026004	044440	041516	051117	
4706	026012	042522	052103	043040	
4707	026020	051117	050040	051117	
4708	026026	020124	043101	042524	
4709	026034	020122	054503	046103	
4710	026042	020105	050125	000	
4711					
4712	026047	122	043505	051511	EM22: .ASCIZ /REGISTER CONTENTS SEEN WHEN DRIVE SWITCHED ON 'OPPOSITE' PORT/
4713	026054	042524	020122	047503	
4714	026062	052116	047105	051524	
4715	026070	051440	042505	020116	
4716	026076	041127	047105	042040	
4717	026104	044522	042526	051440	
4718	026112	044527	041524	042510	
4719	026120	020104	047117	023440	
4720	026126	050117	047520	044523	
4721	026134	042524	020047	047520	
4722	026142	052122	000		
4723					
4724	026145	047	042516	023504	EM23: .ASCIZ /'NED' SET WHEN RPO51 ACCESSED THROUGH PORT NOT SWITCHED/
4725	026152	051440	052105	053440	
4726	026160	042510	020116	050122	
4727	026166	051504	020061	041501	
4728	026174	042503	051523	042105	
4729	026202	052040	051110	052517	
4730	026210	044107	050040	051117	
4731	026216	020124	047516	020124	
4732	026224	053523	052111	044103	
4733	026232	042105	000		
4734					
4735	026235	104	044522	042526	EM24: .ASCIZ /DRIVE SWITCHED TO LOCKED OUT PORT WHEN RELEASED/
4736	026242	051440	044527	041524	
4737	026250	042510	020104	047524	
4738	026256	046040	041517	042513	
4739	026264	020104	052517	020124	
4740	026272	047520	052122	053440	



4741	026300	042510	020116	042522	
4742	026306	042514	051501	042105	
4743	026314	000			
4744					
4745	026315	122	030510	020061	EM25: .ASCIZ /RH11 DIDN'T RESPOND TO ADDRESSING/
4746	026322	044504	047104	052047	
4747	026330	051040	051505	047520	
4748	026336	042116	052040	020117	
4749	026344	042101	051104	051505	
4750	026352	044523	043516	000	
4751					
4752	026357	104	044522	042526	EM30: .ASCIZ /DRIVE NOT SEIZED BY PORT/
4753	026364	047040	052117	051440	
4754	026372	044505	042532	020104	
4755	026400	054502	050040	051117	
4756	026406	000124			
4757					
4758	026410	051127	047117	020107	EM31: .ASCIZ /WRONG STATUS SEEN BY THE SEIZING PORT/
4759	026416	052123	052101	051525	
4760	026424	051440	042505	020116	
4761	026432	054502	052040	042510	
4762	026440	051440	044505	044532	
4763	026446	043516	050040	051117	
4764	026454	000124			
4765					
4766	026456	042522	044507	052123	EM32: .ASCIZ /REGISTER CONTENTS WRONG/
4767	026464	051105	041440	047117	
4768	026472	042524	052116	020123	
4769	026500	051127	047117	000107	
4770					
4771	026506	047503	052116	047522	EM33: .ASCIZ /CONTROL BUS PARITY ERROR READING INDICATED REGISTER/
4772	026514	020114	052502	020123	
4773	026522	040520	044522	054524	
4774	026530	042440	051122	051117	
4775	026536	051040	040505	044504	
4776	026544	043516	044440	042116	
4777	026552	041511	052101	042105	
4778	026560	051040	043505	051511	
4779	026566	042524	000122		
4780					
4781	026572	040503	023516	020124	EM34: .ASCIZ /CAN'T ACCESS DRIVE THROUGH EITHER PORT/
4782	026600	041501	042503	051523	
4783	026606	042040	044522	042526	
4784	026614	052040	051110	052517	
4785	026622	044107	042440	052111	
4786	026630	042510	020122	047520	
4787	026636	052122	000		
4788					
4789	026641	104	044522	042526	EM35: .ASCIZ /DRIVE NOT IN NEUTRAL AFTER RELEASE - REQUEST NOT SET/
4790	026646	047040	052117	044440	
4791	026654	020116	042516	052125	
4792	026662	040522	020114	043101	
4793	026670	042524	020122	042522	
4794	026676	042514	051501	020105	
4795	026704	020055	042522	052521	
4796	026712	051505	020124	047516	

K07

4797	026720	020124	042523	000124	
4798					
4799	026726	051104	053111	020105	EM36: .ASCIZ /DRIVE NOT IN NEUTRAL AFTER TIMEOUT - REQUEST NOT SET/
4800	026734	047516	020124	047111	
4801	026742	047040	052505	051124	
4802	026750	046101	040440	052106	
4803	026756	051105	052040	046511	
4804	026764	047505	052125	026440	
4805	026772	051040	050505	042525	
4806	027000	052123	047040	052117	
4807	027006	051440	052105	000	
4808					
4809	027013	122	043505	051511	EM37: .ASCIZ /REGISTER CONTENTS WRONG AFTER RELEASE OR TIMEOUT/
4810	027020	042524	020122	047503	
4811	027026	052116	047105	051524	
4812	027034	053440	047522	043516	
4813	027042	040440	052106	051105	
4814	027050	051040	046105	040505	
4815	027056	042523	047440	020122	
4816	027064	044524	042515	052517	
4817	027072	000124			
4818					
4819	027074	051104	053111	020105	EM40: .ASCIZ /DRIVE IN NEUTRAL AFTER RELEASE - REQUEST SET/
4820	027102	047111	047040	052505	
4821	027110	051124	046101	040440	
4822	027116	052106	051105	051040	
4823	027124	046105	040505	042523	
4824	027132	026440	051040	050505	
4825	027140	042525	052123	051440	
4826	027146	052105	000		
4827					
4828	027151	122	043505	051511	EM41: .ASCIZ /REGISTER WRONG AFTER RELEASE WITH REQUEST SET/
4829	027156	042524	020122	051127	
4830	027164	047117	020107	043101	
4831	027172	042524	020122	042522	
4832	027200	042514	051101	020105	
4833	027206	044527	044124	051040	
4834	027214	050505	042525	052123	
4835	027222	051440	052105	000	
4836					
4837					
4838	027227	124	051505	020124	DM1: .ASCIZ /TEST # ERR PC PORT # REG ADR CONTENTS/
4839	027234	020043	042440	051122	
4840	027242	050040	020103	050040	
4841	027250	051117	020124	020043	
4842	027256	051040	043505	040440	
4843	027264	051104	041440	047117	
4844	027272	042524	052116	000123	
4845	027300	042524	052123	021440	DM2: .ASCIZ /TEST # ERR PC PORT # REG ADR GOOD BAD/
4846	027306	020040	051105	020122	
4847	027314	041520	020040	047520	
4848	027322	052122	021440	020040	
4849	027330	042522	020107	042101	
4850	027336	020122	047507	042117	
4851	027344	020040	020040	040502	
4852	027352	000104			

4853	027354	042524	052123	021440	DH5:	.ASCIZ	/TEST #	ERR PC	REG ADR	PORT A	PORT B/
4854	027362	020040	051105	020122							
4855	027370	041520	020040	042522							
4856	027376	020107	042101	020122							
4857	027404	047520	052122	040440							
4858	027412	020040	047520	052122							
4859	027420	041040	000								
4860	027423	124	051505	020124	DH6:	.ASCIZ	/TEST #	ERR PC	PORT #/		
4861	027430	020043	042440	051122							
4862	027436	050040	020103	050040							
4863	027444	051117	020124	000043							
4864	027452	042524	052123	021440	DH7:	.ASCIZ	/TEST #	ERR PC	PORT #	TIME (IN MS)/	
4865	027460	020040	051105	020122							
4866	027466	041520	020040	047520							
4867	027474	052122	021440	020040							
4868	027502	044524	042515	024040							
4869	027510	047111	046440	024523							
4870	027516	000									
4871	027517	040	020040	020040	DH13:	.ASCII	/			SEIZE/⟨CR⟩⟨LF⟩	
4872	027524	020040	020040	020040							
4873	027532	020040	020040	051440							
4874	027540	044505	042532	005015							
4875	027546	042524	052123	021440		.ASCIZ	/TEST #	ERR PC	PORT #/		
4876	027554	020040	051105	020122							
4877	027562	041520	020040	047520							
4878	027570	052122	021440	000							
4879	027575	040	020040	020040	DH14:	.ASCII	/			SEIZE ERROR/⟨CR⟩⟨LF⟩	
4880	027602	020040	020040	020040							
4881	027610	020040	020040	051440							
4882	027616	044505	042532	020040							
4883	027624	042440	051122	051117							
4884	027632	005015									
4885	027634	042524	052123	021440		.ASCIZ	/TEST #	ERR PC	PORT #	PORT #	REG ADR CONTENTS/
4886	027642	020040	051105	020122							
4887	027650	041520	020040	047520							
4888	027656	052122	021440	020040							
4889	027664	047520	052122	021440							
4890	027672	020040	042522	020107							
4891	027700	042101	020122	047503							
4892	027706	052116	047105	051524							
4893	027714	000									
4894	027715	124	051505	020124	DH17:	.ASCIZ	/TEST #	ERR PC/			
4895	027722	020043	042440	051122							
4896	027730	050040	000103								
4897	027734	020040	020040	020040	DH24:	.ASCII	/			LOCKED SWITCHED TO/⟨CR⟩⟨LF⟩	
4898	027742	020040	020040	020040							
4899	027750	020040	020040	047514							
4900	027756	045503	042105	020040							
4901	027764	053523	052111	044103							
4902	027772	042105	052040	006517							
4903	030000	012									
4904	030001	124	051505	020124		.ASCIZ	/TEST #	ERR PC	PORT #	PORT #/	
4905	030006	020043	042440	051122							
4906	030014	050040	020103	050040							
4907	030022	051117	020124	020043							
4908	030030	050040	051117	020124							

4909	030036	000043							
4910	030040	051044	040520	051104	DH25:	.ASCIZ	/SRPADR/		
4911	030046	000							
4912	030047	040	020040	020040	DH30:	.ASCII	/	SEIZE	ERROR/<CR><LF>
4913	030054	020040	020040	020040					
4914	030062	020040	020040	051440					
4915	030070	044505	042532	020040					
4916	030076	042440	051122	051117					
4917	030104	005015							
4918	030106	042524	052123	021440		.ASCIZ	/TEST #	ERR PC	PORT # PORT # REG ADR GOOD BAD/
4919	030114	020040	051105	020122					
4920	030122	041520	020040	047520					
4921	030130	052122	021440	020040					
4922	030136	047520	052122	021440					
4923	030144	020040	042522	020107					
4924	030152	042101	020122	047507					
4925	030160	042117	020040	020040					
4926	030166	040502	000104						
4927	030172	020040	020040	020040	DH34:	.ASCII	/	PORT A	PORT B/<CR><LF>
4928	030200	020040	020040	020040					
4929	030206	020040	020040	047520					
4930	030214	052122	040440	020040					
4931	030222	047520	052122	041040					
4932	030230	005015							
4933	030232	042524	052123	021440		.ASCIZ	/TEST #	ERR PC	RPDS1 RPDS1/
4934	030240	020040	051105	020122					
4935	030246	041520	020040	050122					
4936	030254	051504	020061	020040					
4937	030262	050122	051504	000061					
4938	030270	020040	020040	020040	DH35:	.ASCII	/	RELSNG	ERROR/<CR><LF>
4939	030276	020040	020040	020040					
4940	030304	020040	020040	042522					
4941	030312	051514	043516	020040					
4942	030320	051105	047522	006522					
4943	030326	012							
4944	030327	124	051505	020124		.ASCIZ	/TEST #	ERR PC	PORT # PORT #/
4945	030334	020043	042440	051122					
4946	030342	050040	020103	050040					
4947	030350	051117	020124	020043					
4948	030356	050040	051117	020124					
4949	030364	000043							
4950	030366	020040	020040	020040	DH37:	.ASCII	/	RELSNG	ERROR/<CR><LF>
4951	030374	020040	020040	020040					
4952	030402	020040	020040	042522					
4953	030410	051514	043516	020040					
4954	030416	051105	047522	006522					
4955	030424	012							
4956	030425	124	051505	020124		.ASCIZ	/TEST #	ERR PC	PORT # PORT # REG ADR GOOD BAD/
4957	030432	020043	042440	051122					
4958	030440	050040	020103	050040					
4959	030446	051117	020124	020043					
4960	030454	050040	051117	020124					
4961	030462	020043	051040	043505					
4962	030470	040440	051104	043440					
4963	030476	047517	020104	020040					
4964	030504	041040	042101	000					

```

4965 030511 040 020040 020040 DM40: .ASCII / RELSNG RQSTNG/<CR><LF>
4966 030516 020040 020040 020040
4967 030524 020040 020040 051040
4968 030532 046105 047123 020107
4969 030540 051040 051521 047124
4970 030546 006507 012
4971 030551 124 051505 020124 .ASCIZ /TEST # ERR PC PORT # PORT #/
4972 030556 020043 042440 051122
4973 030564 050040 020103 050040
4974 030572 051117 020124 020043
4975 030600 050040 051117 020124
4976 030606 000043
4977
4978 .EVEN
4979
4980 030610 001242 001116 001234 DT1: .WORD TSTNUM, SERRPC, PTNBR, $BDADR, $BDDAT, 0
4981 030616 001122 001126 000000
4982 030624 001242 001116 001234 DT2: .WORD TSTNUM, SERRPC, PTNBR, $BDADR, $GDDAT, $BDDAT, 0
4983 030632 001122 001124 001126
4984 030640 000000
4985 030642 001242 001116 001122 DT5: .WORD TSTNUM, SERRPC, $BDADR, $GDDAT, $BDDAT, 0
4986 030650 001124 001126 000000
4987 030656 001242 001116 001234 DT6: .WORD TSTNUM, SERRPC, PTNBR, 0
4988 030664 000000
4989 030666 001242 001116 001234 DT7: .WORD TSTNUM, SERRPC, PTNBR, TIME, 0
4990 030674 001252 000000
4991 030700 001242 001116 001236 DT13: .WORD TSTNUM, SERRPC, SEIZPT, 0
4992 030706 000000
4993 030710 001242 001116 001236 DT14: .WORD TSTNUM, SERRPC, SEIZPT, PTNBR, $BDADR, $BDDAT, 0
4994 030716 001234 001122 001126
4995 030724 000000
4996 030726 001242 001116 000000 DT17: .WORD TSTNUM, SERRPC, 0
4997 030734 001242 001116 001236 DT24: .WORD TSTNUM, SERRPC, SEIZPT, PTNBR, 0
4998 030742 001234 000000
4999 030746 001272 000000 DT25: .WORD $RPAOR, 0
5000 030752 001242 001116 001236 DT30: .WORD TSTNUM, SERRPC, SEIZPT, PTNBR, $BDADR, $GDDAT, $BDDAT, 0
5001 030760 001234 001122 001124
5002 030766 001126 000000
5003 030772 001242 001116 001170 DT34: .WORD TSTNUM, SERRPC, $TMP2, $TMP3, 0
5004 031000 001172 000000
5005 031004 001242 001116 001236 DT35: .WORD TSTNUM, SERRPC, SEIZPT, PTNBR, 0
5006 031012 001234 000000
5007 031016 001242 001116 001236 DT40: .WORD TSTNUM, SERRPC, SEIZPT, $PPRT, 0
5008 031024 001240 000000
5009
5010 031030 000 000 001 DF1: .BYTE 0,0,1,0,0
5011 031033 000 000
5012 031035 000 000 001 DF2: .BYTE 0,0,1,0,0,0
5013 031040 000 000 000
5014 031043 000 000 000 DF5: .BYTE 0,0,0,0,0
5015 031046 000 000
5016 031050 000 000 001 DF6: .BYTE 0,0,1
5017 031053 000 000 001 DF7: .BYTE 0,0,1,1
5018 031056 001
5019 031057 000 000 001 DF14: .BYTE 0,0,1,1,0,0
5020 031062 001 000 000

```

5021	031065	000	000	DF17:	.BYTE	0,0	
5022	031067	000		DF25:	.BYTE	0	
5023	031070	000	000	001	DF30:	.BYTE	0,0,1,1,0,0,0
5024	031073	001	000	000			
5025	031076	000					
5026	031077	000	000	000	DF34:	.BYTE	0,0,0,0
5027	031102	000					

5028  
5029 031104 .EVEN

5030  
5031  
5032 ;:\*\*\*\*\*

5033  
5034 .SBTTL CONSTANTS, TABLES, ETC

5035  
5036 ;:\*\*\*\*\*

5037  
5038 ;TABLE OF TEST STARTING ADDRESSES

5039				TSTADR:	.WORD	TST1+2	: STARTING ADDRESS OF TEST 1
5040	031104	002714			.WORD	TST2+2	: STARTING ADDRESS OF TEST 2
5041	031106	004314			.WORD	TST3+2	: STARTING ADDRESS OF TEST 3
5042	031110	005276			.WORD	TST4+2	: STARTING ADDRESS OF TEST 4
5043	031112	005660			.WORD	TST5+2	: STARTING ADDRESS OF TEST 5
5044	031114	006576			.WORD	TST6+2	: STARTING ADDRESS OF TEST 6
5045	031116	007514			.WORD	TST7+2	: STARTING ADDRESS OF TEST 7
5046	031120	011072			.WORD	TST10+2	: STARTING ADDRESS OF TEST 10
5047	031122	012450			.WORD	TST11+2	: STARTING ADDRESS OF TEST 11
5048	031124	013362			.WORD	TST12+2	: STARTING ADDRESS OF TEST 12
5049	031126	014274			.WORD	TST13+2	: STARTING ADDRESS OF TEST 13
5050	031130	015312			.WORD	TST14+2	: STARTING ADDRESS OF TEST 14
5051	031132	016424					

5052  
5053 ;ATTENTION BIT TABLE

5054				ATABIT:	.BYTE	1	: ATTENTION BIT FOR DRIVE 0
5055	031134	001			.BYTE	2	: ATTENTION BIT FOR DRIVE 1
5056	031135	002			.BYTE	4	: ATTENTION BIT FOR DRIVE 2
5057	031136	004			.BYTE	10	: ATTENTION BIT FOR DRIVE 3
5058	031137	010			.BYTE	20	: ATTENTION BIT FOR DRIVE 4
5059	031140	020			.BYTE	40	: ATTENTION BIT FOR DRIVE 5
5060	031141	040			.BYTE	100	: ATTENTION BIT FOR DRIVE 6
5061	031142	100			.BYTE	200	: ATTENTION BIT FOR DRIVE 7
5062	031143	200					

5063				MAXTN:	.WORD	\$TN-1	: MAXIMUM TEST NUMBER
5064	031144	000014					
5065							
5066		000001			.END		











MOH = 020000	963#													
MOL = 010000	903#	1764	1796	1868	1893	1978	2003	2105	2148	2250	2293	2446	2465	
	2666	2685	2836	2847	2971	2982	3099	3148	3241	3249	3259	3360	3362	
	3435	3443	3453	3552	3554									
MPE = 000400	860#													
MRO = 000020	933#													
MSE = 000020	992#													
MSTCK = 000010	932#													
MWR = 000040	934#													
MXF = 001000	861#													
NBA = 100000	965#													
NEC = 010000	864#	1668	1672	1689	1693	3305	3309	3499	3503					
NEM = 004000	863#													
NHS = 002000	998#	1016#												
NOATA = 000001	1463#	1922	1928	2032	2038	2179	2185	2324	2330	2456#	2494	2500	2505#	
	2676#	2714	2720	2725#	2876	2882	3011	3017	3130	3135	3179	3184		
	1561	4460#												
NOCLC 023463	1193#													
NOSEIZ 001246	1606	4482#												
NTRH11 023660	1050#	1060#												
OCYL = 100000	1028#													
OFREV = 000200	1024#													
OF100 = 000004	1025#													
OF200 = 000010	1022#													
OF25 = 000001	1026#													
OF400 = 000020	1023#													
OF50 = 000002	1027#													
OF800 = 000040	1058#													
OPE = 020000	923#													
OPT = 020700	1190#	2092*	2237*	5007										
OPRT = 001240	859#													
OR = 000200	913#													
PAR = 000010	856#													
PAT = 000020	862#													
PGE = 002000	900#	1749	1751	1755	1781	1783	1787	1868	1893	1978	2003	2105	2148	
PGM = 001000	2250	2293	2424	2465	2644	2685	2847	2982	3099	3148				
	904#	2424	2644											
PIP = 020000	731#													
PIR0 = 177772	825#													
PIR0VE = 000240	1001#	1018#												
PLU = 020000	1184#	1526*	1527	1531	1534	1538	1550	1553	1659	1660	1705	1706	1743	
PORTA 001224	1744	1811	1851	1852	1884	1885	1894	1913	1915	1921	2004	2023	2025	
	2031	2087	2068	2102	2103	2149	2168	2170	2178	2235	2236	2237	2263	
	2264	2294	2313	2315	2323	2382	2390	2391	2392	2436	2466	2485	2487	
	2493	2525	2526	2602	2669	2670	2686	2705	2707	2713	2728	2729	2769	
	2816	2824	2825	2826	2830	2848	2867	2869	2875	2951	2983	3002	3004	
	3010	3038	3039	3082	3100	3119	3121	3129	3149	3168	3170	3178	3234	
	3235	3320	3321	3323	3358	3359	3476	3477	3518	3519	3563	3564		
	1537	4452#												
PORTAI 023407	1185#	1531*	1532*	1533*	1536*	1544	1680	1681	1722	1723	1775	1776	1813	
PORTB 001226	1898	1909	1910	1927	1961	1962	1994	1995	2008	2019	2020	2037	2090	
	2091	2092	2118	2119	2153	2164	2165	2184	2232	2233	2247	2248	2298	
	2309	2310	2329	2386	2449	2450	2470	2481	2482	2499	2508	2509	2549	
	2606	2610	2611	2612	2656	2690	2701	2702	2719	2745	2746	2820	2852	
	2863	2864	2881	2903	2904	2955	2959	2960	2961	2965	2987	2998	2999	
	3016	3086	3104	3115	3116	3134	3153	3164	3165	3183	3282	3283	3324	









# K08

TST2	004312	1840#	5041											
TST3	005074	1950#	5042											
TST4	005656	2063#	5043											
TST5	006574	2208#	5044											
TST6	007512	2368#	5045											
TST7	011070	2588#	5046											
TUF	= 000100	994#	1012#											
TYPOS	= 104405	3604	3611	3815	4413#									
TYPE	= 104401	1512	1524	1529	1537	1543	1549	1561	1569	1580	1587	1602	1605	1606
		2435	2441	2655	2661	2828	2829	2835	2901	2963	2964	2970	3036	3090
		3091	3139	3140	3190	3236	3243	3244	3355	3356	3357	3376	3430	3437
		3438	3547	3548	3549	3598	3605	3612	3751	3759	3779	3796	3798	3801
		3803	3818	3823	3861	3957	4032	4084	4138	4139	4142	4155	4166	4185
		4233	4239	4244	4248	4253	4254	4256	4259	4263	4328	4330	4409#	
		1604	3787	3811	4141	4410#								
TYPOC	= 104402	4412#												
TYPON	= 104404	1540	1546	2438	2658	2832	2967	4411#						
TYPOS	= 104403	924#												
UNS	= 040000	865#												
UPE	= 020000	852#												
US1	= 000001	853#												
US2	= 000002	854#												
US4	= 000004	1046#												
UWR	= 000010	1045#												
VUF	= 000002	1000#												
VU30	= 010000	897#	1868	1897	1901	1922	1928	1978	2007	2011	2032	2038	2105	2148
VV	= 000100	2152	2156	2250	2293	1922	1928	1978	2007	2011	2032	2038	2105	2148
		2720	2851	2855	2876	2297	1928	1978	2007	2011	2032	2038	2105	2148
		3152	3156			2882	2986	2990	3011	3017	3099	3103	3107	3148
VVSET	= 000001	1463#	1893	1919	1922	1928	2003	2029	2032	2038	2045#	2105	2148	2174
		2176	2179	2180	2185	2186	2250	2293	2319	2321	2324	2325	2330	2331
		2456#	2465	2491	2494	2500	2553#	2676#	2685	2711	2714	2720	2773#	2841#
		2847	2873	2876	2882	2909#	2976#	2982	3008	3011	3017	3044#	3099	3125
		3130	3135	3148	3174	3179	3184							
WAO	= 000002	1055#												
WATCH	001254	1196#	2081#	2122	2226*	2267	2398*	2399	2448*	2453	2553*	2554	2618*	2619
		2668#	2673	2773*	2774	2909*	2910	3044*	3045	3377*	3378	3568*	3569	3661
		3663*	3665*											
WCE	= 040000	866#												
WCF	= 000040	915#												
WCU	= 000001	988#	1006#											
WLE	= 004000	921#												
WRL	= 004000	902#												
WRU	= 000400	996#	1014#											
WSU	= 000004	990#	1008#											
\$AUTO8	001134	1149#	1521#	4131	4279									
\$BOARD	001122	1144#	1664*	1665*	1685*	1686*	1709*	1710*	1726*	1727*	1747*	1748*	1762*	1763*
		1779#	1780*	1794*	1795*	1866*	1867*	1891*	1892*	1976*	1977*	2001*	2002*	2094*
		2095*	2146*	2147*	2239*	2240*	2291*	2292*	2407*	2408*	2422*	2423*	2453*	2454*
		2512*	2513*	2529*	2530*	2627*	2628*	2642*	2643*	2683*	2684*	2732*	2733*	2749*
		2750*	2845*	2846*	2980*	2981*	3097*	3098*	3146*	3147*	3257*	3258*	3286*	3287*
		3301*	3302*	3328*	3329*	3451*	3452*	3480*	3481*	3495*	3496*	3522*	3523*	4980
		4982	4985	4993	5000									
\$BODAT	001126	1146#	1663*	1667	1671	1684*	1688	1692	1708*	1712	1716	1725*	1729	1733
		1746#	1750	1754	1761*	1765	1769	1778*	1782	1786	1793*	1797	1801	1814*
		1815	1865*	1869	1873	1908*	1914*	1920*	1926*	1975*	1979	1983	2018*	2024*

# L08

		2030*	2036*	2093*	2097	2104*	2108	2163*	2169*	2177*	2183*	2238*	2242	2249*
		2253	2308*	2314*	2322*	2328*	2406*	2410	2414	2421*	2425	2429	2480*	2486*
		2492*	2498*	2511*	2515	2519	2528*	2532	2536	2626*	2630	2634	2641*	2645
		2649	2700*	2706*	2712*	2718*	2731*	2735	2739	2748*	2752	2756	2862*	2868*
		2874*	2880*	2997*	3003*	3009*	3015*	3114*	3120*	3128*	3133*	3163*	3169*	3177*
		3182*	3256*	3260	3264	3285*	3289	3293	3300*	3304	3308	3327*	3331	3335
		3450*	3454	3458	3479*	3483	3487	3494*	3498	3502	3521*	3525	3529	4980
		4982	4985	4993	5000									
\$BELL	001202	1172#	3751	3771	4084	4273								
\$CHARC	021062	3863*	3873*	3880	3889*	3894#								
\$CKSWR	022012	4117#	4417											
\$CMTAG	001100	1132#	1475	1476	1484	1488	1489	1490						
\$CM1	= 000001	1164#	1165#											
\$CM2	= 000002	1164#	1165#											
\$CM3	= 000001	1162#	1164											
\$CM4	= 000005	1165#	1166#	1167#	1168#	1169#	1170#							
\$CNTLG	022654	4138	4274#											
\$CNTLU	022647	4155	4248	4273#										
\$CRLF	001207	1174#	1549	1569	1605	2441	2661	2835	2970	3376	3612	3759	3771	3779
		3798	3803	3823	3862	3897	4166	4253	4273	4333				
\$DBLK	021530	3998	4032	4040#										
\$DOAGN	017734	3594	3615	3621#										
\$DTBL	021520	4001	4036#											
\$ENDAD	017724	1111	3617#	3766										
\$ENDCT	017570	1488	3596#											
\$ENULL	017740	3624#												
\$EOP	017524	3572	3584#											
\$EOPCT	017562	1488*	3593#	3597										
\$ERFLG	001103	1135#	2890	2894*	3025	3029*	3344	3348*	3538	3542*	3688	3711	3713*	3730
		3746*	3771											
\$ERMAX	001115	1141#	1491*	1577*	3730									
\$ERROR	020340	1484	3743#											
\$ERRPC	001116	1142#	3753*	3754*	3755	3771	3785	4980	4982	4985	4987	4989	4991	4993
		4996	4997	5000	5003	5005	5007							
\$ERRTB	001276	1227#	3793											
\$ERRTY	020472	3758	3778#											
\$ERTTL	001112	1139#	3609	3613*	3752*	3771								
\$ESCAP	001200	1171#	1490*											
\$FILLC	001156	1160#	3866	3897										
\$FILLS	001155	1159#	3897											
\$GADR	001120	1143#												
\$GODAT	001124	1145#	1666*	1669	1673*	1687*	1690	1694*	1711*	1714	1718*	1728*	1731	1735*
		1749*	1752	1756*	1764*	1767	1771*	1781*	1784	1788*	1796*	1799	1803*	1812*
		1815	1868*	1871	1875*	1893*	1923	1929	1978*	1981	1985*	2003*	2033	2039
		2096*	2097	2105*	2106	2110	2148*	2180	2186	2241*	2242	2250*	2251	2255
		2293*	2325	2331	2409*	2412	2416*	2424*	2427	2431*	2465*	2495	2501	2514*
		2517	2521*	2531*	2534	2538*	2629*	2632	2636*	2644*	2647	2651*	2685*	2715
		2721	2734*	2737	2741*	2751*	2754	2758*	2847*	2877	2883	2982*	3012	3018
		3099*	3130	3135	3148*	3179	3184	3259*	3262	3266*	3288*	3291	3295*	3303*
		3306	3310*	3330*	3333	3337*	3453*	3456	3460*	3482*	3485	3489*	3497*	3500
		3504*	3524*	3527	3531*	4982	4985	5000						
\$GET42	017714	3614#												
\$GTSWR	022102	4139#	4415											
\$HD	= 000000	703												
\$HIOCT	023042	4321*	4332#											
\$ICNT	001104	1136#	1593*	3719*	3720	3722*	3729							











TYPDEC	826	3602	3609	3813										
TYPNAM	826													
TYPNUM	826													
TYPDCS	826	1538	1544	2436	2656	2830	2965							
TYPDCT	826	3785	3810	3810										
TYPDXT	826	3598	3605											
USEREO	3574	3585												
SSCHRE	1125	1164												
SSCHTN	1125	1165	1166	1167	1168	1169								
SSSCA	826													
SSWMT	826	1631	1825	1935	2048	2193	2338	2558	2780	2915	3051	3195	3383	
SSSET	4401	4410	4411	4412	4413	4415	4417	4418	4419	4420	4421	4422		
SSSKIP	826	2913	3048	3381										
.EQUAT	692	716												
.HEADE	692	693												
.SETUP	692	1463												
.SWRHI	692	704												
.SWRLO	692	714												
.SACTI	692	1105												
.SCATC	692	1095												
.SCHTA	692	1125												
.SEOP	692	3574												
.SERRO	692	3730												
.SERRT	692	3771												
.SROOC	692	4280												
.SREAD	692	4041												
.SSAVE	692	4333												
.SSCOP	692	3683												
.STRAP	692	4378												
.STYPD	692	3974												
.STYPE	692	3827												
.STYPO	692	3897												

. ABS. 031146 000

ERRORS DETECTED: 0  
DEFAULT GLOBALS GENERATED: 0

.NOW/SOL/CRF/LI:ME/NL:TOC:MC:MD:CND=DZRJFA.CMB  
RUN-TIME: 75 61 7 SECONDS  
RUN-TIME RATIO: 1068/144=7.3  
CORE USED: 30K (59 PAGES)

