

RM03,02

DUAL PORT LOGIC 2
CZRMHB0

AH-B013B-MC

COPYRIGHT © 1977

FICHE 1 OF 1

JAN 1978

digital

MADE IN USA

EOF1CZRMGBSEQ 00010000 780105 PDP10 411 NTHDR1CZRMHBSEQ 00010000 780105
 00000001 CZRMHBO RMD3/2 DU POR LGC 2 MACY11 30(1046) 21-NOV-77 14:15 PAGE 1
 CZRMHB.P11 21-NOV-77 13:34

IDENTIFICATION

PRODUCT CODE: AC-B012B-MC
PRODUCT NAME: CZRMHBO RM03/RM02 DUAL PORT LOGIC TEST,
PART 2
DATE CREATED: 15 AUG 77
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: DOUG RIIKONEN

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

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

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1977, DIGITAL EQUIPMENT CORPORATION

RMO3 DUAL PORT LOGIC TEST, PART 2

CONTENTS

1. ABSTRACT
2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 PREREQUISITE 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 REQUIRED TESTS
 - 7.5 DISK SURFACE USAGE
 - 7.6 LOOP ON ERROR OPTION
8. TEST DESCRIPTIONS

53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94

1. ABSTRACT

THE RMO3 DUAL PORT LOGIC TEST PERFORMS A SERIES OF TESTS WHICH VERIFY THAT THE RMO3 DUAL PORT LOGIC IS FUNCTIONING PROPERLY. ONLY THE CONTROL LOGIC IS TESTED BY THIS PROGRAM; DATA HANDLING IN THE DUAL PORT 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 PORT LOGIC TO BE TESTED FROM ONE PDP-11/RH11 OR RH70.

THIS PROGRAM IS THE SECOND PART OF THE RMO3 DUAL PORT OPTION LOGIC TEST, AND IS USED TO TEST THE "PORT SELECT" SWITCH.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR
16K OF MEMORY
KW11-L OR KW11-P CLOCK
TELETYPE
RH11 OR RH70 WITH AN RMO3
RMO3 DUAL PORT TEST CABLE

2.2 PREREQUISITE PROGRAMS

A. RMO3 DISKLESS DIAGNOSTIC

B. RMO3 FUNCTIONAL TEST

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

C. RMO3 DUAL PORT LOGIC TEST
PART 1

2.3 OTHER PROGRAMS

DYNAMIC OPERATION OF THE DUAL PORT OPTION IS TESTED BY THE RMO3 PERFORMANCE EXERCISER PROGRAM.

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(8). 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(8). THE PROGRAM WILL USE THE CURRENT DRIVE (DCL) ADDRESS.
- C. THE PROGRAM CAN BE STARTED AT LOCATION 210(8) 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.

MEMORY LOCATION	CONTENTS	FUNCTION
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 PORT 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 'PORT SELECT' SWITCH ON THE DRIVE TO BE TESTED TO THE 'A/B' POSITION. CYCLE THE DRIVE UP.
- D. LOAD THE APPROPRIATE STARTING ADDRESS (200(8) OR 210(8)) INTO THE SWITCH REGISTER (OR THE 'SOFTWARE' SWITCH REGISTER, SEE SECTION 5.2.)
- E. PRESS START.
- F. ENTER THE DRIVE NUMBER.
- 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 RMO3 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

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

207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262

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'.

5.4

TEST CABLE CONNECTION

TO TEST THE RMO3 DUAL PORT 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 RMO3 WILL RESPOND TO A DIFFERENT MASSBUS ADDRESS.

THE ADDRESS OF EACH PORT WILL DEPEND UPON THE DRIVE'S ADDRESS PLUG.

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/RM11 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 BITS FOR PORTS A & B ARE ASSERTED IN THE SAME BIT POSITION WHEN 'RMAS' (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 'RMDSI' (DRIVE STATUS REGISTER) TO DETERMINE THE STATE OF THE SELECTED PORT'S ATTENTION BIT.

263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318

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 TYPEOUT 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 RMO3 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 PORT OPTION.

7.3 EXECUTION TIME

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

7.4 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 - 10 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 - 10 ARE RUN.

7.5 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 RMO3 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.

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 (RMD51) 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 RMD51, 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 (RMD51) THROUGH THE SEIZING PORT AND VERIFING THAT CORRECT STATUS IS SEEN. WHEN RMD51 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.

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 RMO3, THAT THE DRIVE IS ONLINE (RMD51 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.

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 BIT IS SET.

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.

TEST 4 MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT A

MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT A

- A. WRITE 0'S INTO RMDS1 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

TEST 5 MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT B

MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT B

- A. WRITE 0'S INTO RMDS1 THROUGH PORT B AND VERIFY THAT THE DRIVE HAS BEEN SEIZED.

431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486

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

TEST 6 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 RMDS1, 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 RMDS1, AS READ THROUGH BOTH
PORTS, ARE CORRECT.

C. RETURN THE 'CONTROLLER SELECT' SWITCH TO THE 'A/B' POSITION. VERIFY
THE DRIVE STATE.

TEST 7 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 RMDS1, 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' SETS WHEN ATTEMPTING TO ACCESS THE DRIVE THROUGH
PORT B. ATTEMPT TO SET PORT REQUEST BY WRITING 0'S
INTO RMDS1 THROUGH PORT B.

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

487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542

TEST 10 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 RMDS1, 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' SETS WHEN ATTEMPTING TO ACCESS THE DRIVE THROUGH PORT A. ATTEMPT TO SET PORT REQUEST BY WRITING 0'S INTO RMDS1 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.

↑

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

;PROGRAM REVISION #001

.TITLE CZRMH80 RM03/2 DU POR LGC 2
;*COPYRIGHT (C) 1977
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY D. RIIKONEN
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
;*

.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 EMT,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= 13	;;CODE FOR CARRIAGE RETURN
CRLF= 200	;;CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776	;;PROCESSOR STATUS WORD
.EQUIV PS,PSW	
STKLMT= 177774	;;STACK LIMIT REGISTER
PIRQ= 177772	;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570	;;HARDWARE SWITCH REGISTER
DDISP= 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

001100

000011
000012
000015
000200
177776

177774
177772
177570
177570

000000
000001
000002
000003
000004
000005
000006
000007
000006

635 000007

PC= %7 ;;PROGRAM COUNTER

636
637
638 000000
639 000040
640 000100
641 000140
642 000200
643 000240
644 000300
645 000340

;;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

646
647
648 100000
649 040000
650 020000
651 010000
652 004000
653 002000
654 001000
655 000400
656 000200
657 000100
658 000040
659 000020
660 000010
661 000004
662 000002
663 000001

;;"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
SW09,SW9
SW08,SW8
SW07,SW7
SW06,SW6
SW05,SW5
SW04,SW4
SW03,SW3
SW02,SW2
SW01,SW1
SW00,SW0

664
665
666
667
668
669
670
671
672
673
674
675
676 100000
677 040000
678 020000
679 010000
680 004000
681 002000
682 001000
683 000400
684 000200
685 000100
686 000040
687 000020
688 000010
689 000004
690 000002

;;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

```

691          000001      BIT00= 1
692          .EQUIV      BIT09,BIT9
693          .EQUIV      BIT08,BIT8
694          .EQUIV      BIT07,BIT7
695          .EQUIV      BIT06,BIT6
696          .EQUIV      BIT05,BIT5
697          .EQUIV      BIT04,BIT4
698          .EQUIV      BIT03,BIT3
699          .EQUIV      BIT02,BIT2
700          .EQUIV      BIT01,BIT1
701          .EQUIV      BIT00,BIT0
702
703          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
704          ERRVEC= 4      ;TIME OUT AND OTHER ERRORS
705          RESVEC= 10     ;RESERVED AND ILLEGAL INSTRUCTIONS
706          TBITVEC= 14    ;"T" BIT
707          TRTVEC= 14     ;TRACE TRAP
708          BPTVEC= 14     ;BREAKPOINT TRAP (BPT)
709          IOTVEC= 20     ;INPUT/OUTPUT TRAP (IOT) **SCOPE**
710          PWRVEC= 24     ;POWER FAIL
711          EMTVEC= 30     ;EMULATOR TRAP (EMT) **ERROR**
712          TRAPVEC= 34    ;"TRAP" TRAP
713          TKVEC= 60      ;TTY KEYBOARD VECTOR
714          TPVEC= 64     ;TTY PRINTER VECTOR
715          PIRQVEC= 240   ;PROGRAM INTERRUPT REQUEST VECTOR
716
717          ;;*****
718          ;;*****
719
720          .SBTTL  RH11 REGISTERS
721
722          ;;*****
723
724          ;CONTROL AND STATUS REGISTER 1 (RMCS1)
725
726          IE= 100        ;INTERRUPT ENABLE (BIT #6)
727          RDY= 200       ;READY (BIT #7)
728          A16= 400       ;HIGH ORDER BUS ADDRESS BIT (BIT #8)
729          A17= 1000      ;HIGH ORDER BUS ADDRESS BIT (BIT #9)
730          PSEL= 2000     ;PORT SELECT (BIT #10)
731          MCPE= 20000    ;MASSBUSS PARITY ERROR (BIT #13)
732          TRE= 40000     ;TRANSFER ERROR (BIT #14)
733          SC= 100000     ;SPECIAL CONDITION (BIT #15)
734
735          ;WORD COUNT REGISTER (RMWC)
736          ;(EACH BIT IS CALLED BY BIT NUMBER)
737
738          ;BUS ADDRESS REGISTER (RMBA)
739          ;(EACH BIT IS CALLED BY BIT NUMBER)
740
741          ;CONTROL AND STATUS REGISTER 2 (RMCS2)
742
743          U0= 1           ;UNIT SELECT (BIT #0)
744          U1= 2           ;UNIT SELECT (BIT #1)
745          U3= 4           ;UNIT SELECT (BIT #2)
746

```


747	000010	BAI=	10	;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
748	000020	PAT=	20	;MASSBUS PARITY TEST (BIT #4)
749	000040	CLR=	40	;CLEAR (BIT #5)
750	000100	IR=	100	;INPUT READY (BIT #6)
751	000200	OR=	200	;OUTPUT READY (BIT #7)
752	000400	MDPE=	400	;MASS BUS PARITY ERROR (BIT #8)
753	001000	MXF=	1000	;MISSED TRANSFER ERROR (BIT #9)
754	002000	PGE=	2000	;PROGRAM ERROR (BIT #10)
755	004000	NEM=	4000	;NON EXISTENT MEMORY (BIT #11)
756	010000	NED=	10000	;NON EXISTENT DRIVE (BIT #12)
757	020000	UPE=	20000	;UNIBUS PARITY ERROR (BIT #13)
758	040000	WCE=	40000	;WRITE CHECK ERROR (BIT #14)
759	100000	DLT=	100000	;DATA LATE (BIT #15)
760				
761				;DATA BUFFER REGISTER (RMD8)
762				; (EACH BIT IS CALLED BY BIT NUMBER)
763				
764				
765				;;*****
766				
767				.SBTTL RMO3 REGISTERS
768				;;*****
769				
770				;CONTROL AND STATUS 1 REGISTER. (#00)
771				
772				
773	000001	GO=	1	;GO BIT (BIT #0)
774	000002	FO=	2	;FUNCTION CODE BIT #1
775	000004	F1=	4	;FUNCTION CODE BIT #2
776	000010	F2=	10	;FUNCTION CODE BIT #3
777	000020	F3=	20	;FUNCTION CODE BIT #4
778	000040	F4=	40	;FUNCTION CODE BIT #5
779	004000	DVA=	4000	;DEVICE AVAILABLE (BIT #11)
780				
781				;DRIVE STATUS REGISTER (RMDS1) (#01)
782				
783				
784	000002	DFS=	1	DRIVE FORWARD 5"/SEC. (BIT #0)
785	000004	DFF20=	2	;DRIVE FORWARD 20"/SEC. (BIT #1)
786	000010	DIG8=	4	;DRIVE TO INNER GUARD BAND (BIT #2)
787	000020	GRV=	10	;GO REVERSE (BIT #3)
788	000040	DL64=	20	;DIFFERENCE LESS THAN 64 (BIT #4)
789	000100	DE1=	40	;DIFFERENCE EQUALS 1 (BIT #5)
790	000200	VV=	100	;VOLUME VALID (BIT #6)
791	000400	DRY=	200	;DRIVE READY (BIT #7)
792	001000	DPR=	400	;DRIVE PRESENT (BIT #8)
793	002000	PGM=	1000	;PROGRAMABLE (BIT #9)
794	004000	LBT=	2000	;LAST SECTOR TRANSFERRED (BIT #10)
795	010000	WRL=	4000	;WRITE LOCK (BIT #11)
796	020000	MOL=	10000	;MEDIUM ON-LINE (BIT #12)
797	040000	PIP=	20000	;POSITIONING OPERATION IN PROGRESS (BIT #13)
798	100000	ERR=	40000	;COMPOSITE ERROR (BIT #14)
799		ATA=	100000	;ATTENTION ACTIVE (BIT #15)
800				
801				;ERROR REGISTER #01 (RMER1) (#02)
802	000001	ILF=	1	;ILLEGAL FUNCTION (BIT #0)

803	000002	ILR=	2	; ILLEGAL REGISTER (BIT #1)
804	000004	RMR=	4	; REGISTER MODIFICATION REFUSED (BIT #2)
805	000010	PAR=	10	; PARITY ERROR (BIT #3)
806	000020	FER=	20	; FORMAT ERROR (BIT #4)
807	000040	WCF=	40	; WRITE CLOCK FAIL (BIT #5)
808	000100	ECH=	100	; ECC HARD ERROR (BIT #6)
809	000200	HCE=	200	; HEADER COMPARE ERROR (BIT #7)
810	000400	HCRC=	400	; HEADER CRC ERROR (BIT #8)
811	001000	AOE=	1000	; ADDRESS OVERFLOW ERROR (BIT #9)
812	002000	IAE=	2000	; INVALID ADDRESS ERROR (BIT #10)
813	004000	WLE=	4000	; WRITE LOCK ERROR (BIT #11)
814	010000	DTE=	10000	; DRIVE TIMING ERROR (BIT #12)
815	020000	OPI=	20000	; OPERATION INCOMPLETE (BIT #13)
816	040000	UNS=	40000	; DRIVE UNSAFE (BIT #14)
817	100000	DCK=	100000	; DATA CHECK ERROR (BIT #15)
818				
819				
820				
821	000001	DMD=	1	; DIAGNOSTIC MODE (BIT #0)
822				
823				
824				
825	000001	ATO=	1	; DEVICE 0 (BIT #0)
826	000002	AT1=	2	; DEVICE 1 (BIT #1)
827	000004	AT2=	4	; DEVICE 2 (BIT #2)
828	000010	AT3=	10	; DEVICE 3 (BIT #3)
829	000020	AT4=	20	; DEVICE 4 (BIT #4)
830	000040	AT5=	40	; DEVICE 5 (BIT #5)
831	000100	AT6=	100	; DEVICE 6 (BIT #6)
832	000200	AT7=	200	; DEVICE 7 (BIT #7)
833				
834				
835				
836				
837				
838				
839	000001	DT00=	1	; DRIVE TYPE NUMBER BIT 1
840	000002	DT01=	2	; DRIVE TYPE NUMBER BIT 2
841	000004	DT02=	4	; DRIVE TYPE NUMBER BIT 3
842	000010	DT03=	10	; DRIVE TYPE NUMBER BIT 4
843	000020	DT04=	20	; DRIVE TYPE NUMBER BIT 5
844	000040	DT05=	40	; DRIVE TYPE NUMBER BIT 6
845	000100	DT06=	100	; DRIVE TYPE NUMBER BIT 7
846	000200	DT07=	200	; DRIVE TYPE NUMBER BIT 8
847	000400	DT08=	400	; DRIVE TYPE NUMBER BIT 9
848	004000	DRQ=	4000	; DRIVE REQUEST REQUIRED (BIT #11)
849	020000	MOH=	20000	; MOVING HEAD (BIT #13)
850	040000	TAP=	40000	; TAPE DRIVE (BIT #14)
851	100000	NBA=	100000	; NOT BLOCK ADDRESSED (BIT #15)
852				
853				
854				
855	000100	SC0=	100	; SECTOR COUNT FIELD 0 (BIT #6)
856	000200	SC1=	200	; SECTOR COUNT FIELD 1 (BIT #7)
857	000400	SC2=	400	; SECTOR COUNT FIELD 2 (BIT #8)
858	001000	SC3=	1000	; SECTOR COUNT FIELD 3 (BIT #9)

859	002000	SC4= 2000	;SECTOR COUNT FIELD 4 (BIT #10)
860			
861		;RM03 ERROR REGISTER #2 (RMER2) (#10)	
862			
863	000010	DPE= 10	;DATA PARITY ERROR (BIT #3)
864	000200	DVC= 200	;DEVICE CHECK (BIT #7)
865	002000	LBC= 2000	;LOSS OF BIT CLOCK (BIT #10)
866	004000	LSC= 4000	;LOSS OF SYSTEM CLOCK (BIT #11)
867	010000	IVC= 10000	;INVALID COMMAND (BIT #12)
868	020000	OPE= 20000	;OPERATOR ERROR (BIT #13)
869	100000	SKI= 100000	;SEEK INCOMPLETE (BIT #14)
870			
871		;OFFSET REGISTER (RMOF) (#11)	
872			
873	000200	OFD= 200	;OFFSET FORWARD (BIT #5)
874	002000	HCI= 2000	;HEADER COMPARE INHIBIT (BIT #10)
875	004000	ECI= 4000	;ERROR CORRECTION CODE INHIBIT (BIT #11)
876	010000	FMT16= 10000	;FORMAT BIT (BIT #12)
877			
878		;DESIRED CYLINDER ADDRESS (RMDC) (#12)	
879		; (EACH BIT IS CALLED BY BIT NUMBER)	
880			
881		;SERIAL NUMBER REGISTER (RMSN) (#14)	
882		; (EACH IS CALLED BY BIT NUMBER)	
883			
884		;ECC POSITION REGISTER (RMEC1) (#16)	
885		; (EACH BIT IS CALLED BY BIT NUMBER)	
886			
887		;ECC PATTERN REGISTER (RMEC2) (#17)	
888		; (EACH BIT IS CALLED BY BIT NUMBER)	
889			
890		;;*****	
891			
892		.SBTTL DEFINITIONS OF THE RH11/RM03 ADDRESS INDEXES	
893			
894		;;*****	
895			
896	000000	RMCS1=0	;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
897	000002	RMWC=2	;WORD COUNT REGISTER (NOT A DRIVE REG)
898	000004	RMBA=4	;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
899	000006	RMDA=6	;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
900	000010	RMCS2=10	;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
901	000012	RMDS1=12	;DRIVE STATUS REGISTER (DRIVE REG 01)
902	000014	RMER1=14	;ERROR REGISTER #1 (DRIVE REG. 02)
903	000016	RMA5=16	;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
904	000020	RMLA=20	;LOOK AHEAD REGISTER (DRIVE REG. 07)
905	000022	RMD8=22	;DATA BUFFER REGISTER (NOT A DRIVE REG.)
906	000024	RMMR1=24	;MAINTAINABILITY REGISTER (DRIVE REG. 03)
907	000026	RMDT=26	;DRIVE TYPE REGISTER (DRIVE REG. 06)
908	000030	RMSN=30	;SERIAL NUMBER REGISTER (DRIVE REG. 10)
909	000032	RMOF=32	;OFFSET REGISTER (DRIVE REG. 11)
910	000034	RMDC=34	;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
911	000040	RMMR2=40	;MAINTENANCE REGISTER #2 (DRIVE REG. 14)
912	000042	RMER2=42	;ERROR REGISTER #2 (DRIVE REG. 15)
913	000044	RMEC1=44	;ECC POSITION REGISTER (DRIVE REG. 16)
914	000046	RMEC2=46	;ECC PATTERN REGISTER (DRIVE REG. 17)

CZRMHBO RM03/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34

MACY11 30(1046) 21-NOV-77 14:15 PAGE 18
DEFINITIONS OF THE RH11/RM03 ADDRESS INDEXES

SEQ 0018

```

915
916
917
918          000000
919
920          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A "+2,HALT"
921          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
922          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
923          ;*=0
924          DISPREG: .WORD 0          ;; SOFTWARE DISPLAY REGISTER
925          SWREG:   .WORD 0          ;; SOFTWARE SWITCH REGISTER
926
927          .SBTTL  ACT11 HOOKS
928          ;*****
929          ;HOOKS REQUIRED BY ACT11
930          SSVPC=.          ;SAVE PC
931          .=46
932          SENDAD          ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
933          .=52
934          .WORD 20000      ;;2)SET LOC.52 TO 20000
935          .=$SVPC          ;; RESTORE PC
936
937          .SBTTL  STARTING ADDRESS = 200
938          JMP     START          ;START THE PROGRAM
939          000200 000137 001706
940
941          .SBTTL  START THE PROGRAM AND CHANGE THE RH11 ADDRESS = 204
942          JMP     START1         ;START AND CHANGE THE RH11 ADDRESS
943          000204 000137 001714
944
945

```


.SBTTL COMMON TAGS

```

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

```

```

946
947
948
949
950
951
952 001100
953 001100
954 001100 000000
955 001102 000
956 001103 000
957 001104 000000
958 001106 000000
959 001110 000000
960 001112 000000
961 001114 000
962 001115 001
963 001116 000000
964 001120 000000
965 001122 000000
966 001124 000000
967 001126 000000
968 001130 000000
969 001132 000000
970 001134 000
971 001135 000
972 001136 000000
973 001140 177570
974 001142 177570
975 001144 177560
976 001146 177562
977 001150 177564
978 001152 177566
979 001154 000
980 001155 002
981 001156 012
982 001157 000
983 001160 000000
984
985 001162 000000
986 001164 000000
987 001166 000000
988 001170 000000
989 001172 000000
990 001174 000000
991 001176 000000
992 001200 000000
993 001202 177607 000377
994 001206 077
995 001207 015
996 001210 000012
997
998 000015
999 000012
1000 001212 172540
1001 001214 172542

```

```

.=1100
SCMTAG: .WORD 0
SPASS: .WORD 0
STSTNM: .BYTE 0
SERFLG: .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
$GDOAT: .WORD 0
$BDOAT: .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: .ASCIZ <207><377><377>
$QUES: .ASCIZ /?/
$CRLF: .ASCIZ <15>
$LF: .ASCIZ <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 ($REGO) WAS OBTAINED
;CONTAINS (($REGAD)+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

```

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34

MACY11 30(1046) 21-NOV-77 14:15 PAGE 20
COMMON TAGS

SEQ 0020

1002	001216	000104	\$LPVEC: .WORD	104	; ADDR OF KW11-P VECTOR
1003	001220	177546	\$LKS: .WORD	177546	; ADDR OF KW11-L STATUS REGISTER
1004	001222	000100	\$LLVEC: .WORD	100	; ADDR OF KW11-L VECTOR
1005	001224	000000	PORTA: .WORD	0	; ADDRESS OF PORT A
1006	001226	000000	PORTB: .WORD	0	; ADDRESS OF PORT B
1007	001230	000000	PORTC: .WORD	0	; ADDRESS OF DIFFERENT DRIVE
1008	001232	000000	ASR1: .WORD	0	; ATA-A OR ATA-B = 1
1009	001234	000000	PTNR: .WORD	0	; CONTAINS THE PORT ADDRESS FOR ERROR TYPEOUTS
1010	001236	000000	SEIZPT: .WORD	0	; CONTAINS THE ADDRESS OF THE SEIZING PORT
1011	001240	000000	OPPR: .WORD	0	; CONTAINS THE ADDRESS OF THE 'OPPOSITE' PORT
1012	001242	000000	TSTNUM: .WORD	0	; NUMBER OF THE CURRENT TEST
1013	001244	000000	CKERR: .WORD	0	; IF -1, A REGISTER MISCOMPARISON OCCURRED
1014	001246	000000	NOSEIZ: .WORD	0	; IF -1, THE PORT IN 'SEIZPT' DID NOT SEIZE THE DRIVE
1015	001250	000000	RELERR: .WORD	0	; IF -1, THE PORT IN 'SEIZPT' DID NOT RELEASE THE DRIVE
1016	001252	000000	TIME: .WORD	0	; ELAPSED TIME COUNTER
1017	001254	000000	WATCH: .WORD	0	; WATCH DOG TIMER LOCATION
1018	001256	000000	TIMEA: .WORD	0	; THE TIMEOUT ONE-SHOT VALUE MEASURED THROUGH PORT A
1019	001260	000000	TIMEAP: .WORD	0	; PORT A TIMEOUT VALUE + 25%
1020	001262	000000	TIMEB: .WORD	0	; THE TIMEOUT ONE-SHOT VALUE MEASURED THROUGH PORT B
1021	001264	000000	TIMEBP: .WORD	0	; PORT B TIMEOUT VALUE + 25%
1022	001266	000000	KYBCTL: .WORD	0	; SINGLE TEST INDICATOR
1023	001270	000000	CHGADR: .WORD	0	; CHANGE THE RH11 ADDRESS INDICATOR
1024					
1025					
1026					
1027					
1028					
1029					
1030					
1031	001272	176700	\$RMADR: .WORD	176700	; RH11/RM03 UNIBUS ADDRESS
1032	001274	000254	\$RMVEC: .WORD	254	; RH11 INTERRUPT VECTOR ADDRESS
1033					

.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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
 ;*NOTE1: IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

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

\$ERRTB:

;ERROR 1

EM1	;DRIVE IS NON-EXISTENT ('NED' BIT SET)
DH1	
DT1	
DF1	

;ERROR 2

EM2	;WRONG DRIVE TYPE
DH2	
DT2	
DF2	

;ERROR 3

EM3	;CONTROLLER SELECT SWITCH ON DRIVE NOT IN 'A/B'
DH1	
DT1	
DF1	

;ERROR 4

EM4	;DRIVE NOT ON LINE
DH2	
DT2	
DF2	

;ERROR 5

EM5	;SERIAL NUMBER READ THROUGH EACH PORT NOT THE SAME
DH5	
DT5	
DF5	

;ERROR 6

EM6	;TIMEOUT HAS NOT OCCURRED WITHIN 2 SECONDS
DH6	

1034		
1035		
1036		
1037		
1038		
1039		
1040		
1041		
1042		
1043		
1044		
1045		
1046		
1047		
1048	001276	
1049		
1050		
1051		
1052		
1053	001276	017747
1054	001300	022523
1055	001302	024104
1056	001304	024324
1057		
1058		
1059		
1060	001306	020015
1061	001310	022574
1062	001312	024120
1063	001314	024331
1064		
1065		
1066		
1067	001316	020036
1068	001320	022523
1069	001322	024104
1070	001324	024324
1071		
1072		
1073		
1074	001326	020115
1075	001330	022574
1076	001332	024120
1077	001334	024331
1078		
1079		
1080		
1081	001336	020137
1082	001340	022650
1083	001342	024136
1084	001344	024337
1085		
1086		
1087		
1088	001346	020221
1089	001350	022717

1090	001352	024152	DT6	
1091	001354	024344	DF6	
1092				
1093				; ERROR 7
1094				
1095	001356	020273	EM7	; TIMEOUT ONE-SHOT IS LESS THAN 500 MS
1096	001360	022746	DH7	
1097	001362	024162	DT7	
1098	001364	024347	DF7	
1099				
1100				; ERROR 10
1101				
1102	001366	020340	EM10	; READIN PRESET DOES NOT SET VOLUME VALID FOR THE PORT
1103	001370	022523	DH1	
1104	001372	024104	DT1	
1105	001374	024324	DF1	
1106				
1107				; ERROR 11
1108				
1109	001376	020425	EM11	; 'GO' BIT RESET DURING UNLOAD COMMAND
1110	001400	022523	DH1	
1111	001402	024104	DT1	
1112	001404	024324	DF1	
1113				
1114				; ERROR 12
1115				
1116	001406	020472	EM12	; INCORRECT STATUS DURING UNLOAD COMMAND
1117	001410	022523	DH1	
1118	001412	024104	DT1	
1119	001414	024324	DF1	
1120				
1121				; ERROR 13
1122				
1123	001416	020541	EM13	; DRIVE DID NOT RETURN TO NEUTRAL AFTER UNLOAD COMMAND
1124	001420	023013	DH13	
1125	001422	024174	DT13	
1126	001424	024344	DF6	
1127				
1128				; ERROR 14
1129				
1130	001426	020626	EM14	; ATTENTION BIT SET ON 'OPPOSITE PORT' AFTER UNLOAD
1131	001430	023071	DH14	
1132	001432	024204	DT14	
1133	001434	024353	DF14	
1134				
1135				; ERROR 15
1136				
1137	001436	020710	EM15	; ATTENTION BIT NOT SET ON PORT WHICH ISSUED 'UNLOAD'
1138	001440	022523	DH1	
1139	001442	024104	DT1	
1140	001444	024324	DF1	
1141				
1142				; ERROR 16
1143				
1144	001446	020774	EM16	; DRIVE NOT IN NEUTRAL AFTER UNLOAD WITH 'CONTROLLER
1145				; SELECT' SWITCH MOVED FROM 'A/B'

K02

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 23
ERROR POINTER TABLE

SEQ 0023

1146	001450	023013	DH13	
1147	001452	024174	DT13	
1148	001454	024344	DF6	
1149				;ERROR 17
1150				
1151	001456	021120	EM17	;DRIVE LOCKED ON PORT 'A' BY SWITCH WHILE CYCLED UP
1152	001460	023211	DH17	
1153	001462	024222	DT17	
1154	001464	024361	DF17	
1155				
1156				;ERROR 20
1157				
1158	001466	021203	EM20	;DRIVE LOCKED ON PORT 'B' BY SWITCH WHILE CYCLED UP
1159	001470	023211	DH17	
1160	001472	024222	DT17	
1161	001474	024361	DF17	
1162				
1163				;ERROR 21
1164				
1165	001476	021266	EM21	;STATUS INCORRECT FOR PORT AFTER CYCLE UP
1166	001500	022523	DH1	
1167	001502	024104	DT1	
1168	001504	024324	DF1	
1169				
1170				;ERROR 22
1171				
1172	001506	021337	EM22	;REGISTER CONTENTS SEEN WHEN DRIVE SWITCHED ON 'OPPOSITE' PORT
1173	001510	022523	DH1	
1174	001512	024104	DT1	
1175	001514	024324	DF1	
1176				
1177				;ERROR 23
1178				
1179	001516	021435	EM23	; 'NED' SET WHEN RMDS1 ACCESSED THROUGH PORT NOT SWITCHED
1180	001520	022523	DH1	
1181	001522	024104	DT1	
1182	001524	024324	DF1	
1183				
1184				;ERROR 24
1185				
1186	001526	021531	EM24	;DRIVE SWITCHED TO LOCKED OUT PORT WHEN RELEASED/
1187	001530	023230	DH24	
1188	001532	024230	DT24	
1189	001534	024347	DF7	
1190				
1191				;ERROR 25
1192				
1193	001536	021611	EM25	;RH11 DIDN'T RESPOND TO ADDRESSING
1194	001540	023334	DH25	
1195	001542	024242	DT25	
1196	001544	024363	DF25	
1197				
1198				;ERROR 26
1199				
1200	001546	000000	0	;UNUSED ERROR MESSAGES
1201	001550	000000	0	

1202	001552	000000	0	
1203	001554	000000	0	
1204				
1205				; ERROR 27
1206				
1207	001556	000000	0	; UNUSED ERROR MESSAGES
1208	001560	000000	0	
1209	001562	000000	0	
1210	001564	000000	0	
1211				
1212				; ERROR 30
1213				
1214	001566	021653	EM30	; DRIVE NOT SEIZED BY PORT 'N'
1215	001570	023343	DH30	
1216	001572	024246	DT30	
1217	001574	024364	DF30	
1218				
1219				; ERROR 31
1220				
1221	001576	021704	EM31	; WRONG STATUS SEEN BY THE SEIZING PORT
1222	001600	022574	DH2	
1223	001602	024120	DT2	
1224	001604	024331	DF2	
1225				
1226				; ERROR 32
1227				
1228	001606	021752	EM32	; REGISTER CONTENTS INCORRECT
1229	001610	022574	DH2	
1230	001612	024120	DT2	
1231	001614	024331	DF2	
1232				
1233				; ERROR 33
1234				
1235	001616	022002	EM33	; CONTROL BUS PARITY ERROR WHILE READING REGISTER
1236	001620	022523	DH1	
1237	001622	024104	DT1	
1238	001624	024324	DF1	
1239				
1240				; ERROR 34
1241				
1242	001626	022066	EM34	; CAN'T ACCESS DRIVE THROUGH EITHER PORT
1243	001630	023466	DH34	
1244	001632	024266	DT34	
1245	001634	024373	DF34	
1246				
1247				; ERROR 35
1248				
1249	001636	022135	EM35	; DRIVE NOT IN NEUTRAL AFTER RELEASE, REQUEST NOT SET
1250	001640	023564	DH35	
1251	001642	024300	DT35	
1252	001644	024347	DF7	
1253				
1254				; ERROR 36
1255				
1256	001646	022222	EM36	; DRIVE NOT IN NEUTRAL AFTER TIMEOUT, REQUEST NOT SET
1257	001650	023564	DH35	

1258	001652	024300		DT35	
1259	001654	024347		DF7	
1260					
1261					
1262					
1263	001656	022307		EM37	;REGISTER CONTENTS INCORRECT AFTER RELEASE/TIMEOUT
1264	001660	023662		DH37	
1265	001662	024246		DT30	
1266	001664	024364		DF30	
1267					
1268					
1269					
1270	001666	022370		EM40	;DRIVE NOT SEIZED BY PORT AFTER RELEASE WITH REQUEST SET
1271	001670	024005		DH40	
1272	001672	024312		DT40	
1273	001674	024347		DF7	
1274					
1275					
1276					
1277	001676	022445		EM41	;REGISTER WRONG AFTER RELEASE WITH REQUEST SET
1278	001700	023343		DH30	
1279	001702	024246		DT30	
1280	001704	024364		DF30	
1281					
1282					
1283					
1284					
1285					
1286					
1287					
1288					
1289					
1290					
1291	001706	005037	001270	START: CLR	CHGADR ;CLEAR THE 'CHANGE RH11 ADDRESS' INDICATOR
1292	001712	000403		BR	START2 ;GO TO THE START
1293	001714	012737	177777 001270	START1: MOV	#-1,CHGADR ;SET THE 'CHANGE RH11 ADDRESS' INDICATOR
1294	001722	000005		START2: RESET	;CLEAR THE BUS
1295				.SBTTL	INITIALIZE THE COMMON TAGS
1296				;;CLEAR	THE COMMON TAGS (\$CHTAG) AREA
1297	001724	012706	001100	MOV	\$CHTAG,R6 ;FIRST LOCATION TO BE CLEARED
1298	001730	005026		CLR	(R6)+ ;CLEAR MEMORY LOCATION
1299	001732	022706	001140	CMP	\$SWR,R6 ;;DONE?
1300	001736	001374		BNE	-6 ;;LOOP BACK IF NO
1301	001740	012706	001100	MOV	\$STACK,SP ;;SETUP THE STACK POINTER
1302				;;INITIALIZE A FEW VECTORS	
1303	001744	012737	013460 000020	MOV	\$SCOPE,\$IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1304	001752	012737	000340 000022	MOV	\$340,\$IOTVEC+2 ;;LEVEL 7
1305	001760	012737	013642 000030	MOV	\$ERROR,\$EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1306	001766	012737	000340 000032	MOV	\$340,\$EMTVEC+2 ;;LEVEL 7
1307	001774	012737	016442 000034	MOV	\$TRAP,\$TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1308	002002	012737	000340 000036	MOV	\$340,\$TRAPVEC+2 ;;LEVEL 7
1309	002010	013737	013072 013064	MOV	\$ENDCT,\$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
1310	002016	005037	001176	CLR	\$TIMES ;;INITIALIZE NUMBER OF ITERATIONS
1311	002022	005037	001200	CLR	\$ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1312	002026	112737	000001 001115	MOVB	#1,\$SERMAX ;;ALLOW ONE ERROR PER TEST
1313	002034	012737	002034 001106	MOV	#, \$LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE

1314	002042	012737	002042	001110	MOV	#,SLPERR	;;SETUP THE ERROR LOOP ADDRESS
1315					;;SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS		
1316					;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.		
1317	002050	013746	000004		MOV	#ERRVEC, -(SP)	;;SAVE ERROR VECTOR
1318	002054	012737	002110	000004	MOV	#64\$, #ERRVEC	;;SET UP ERROR VECTOR
1319	002062	012737	177570	001140	MOV	#DSWR, SWR	;;SETUP FOR A HARDWARE SWICH REGISTER
1320	002070	012737	177570	001142	MOV	#DDISP, DISPLAY	;;AND A HARDWARE DISPLAY REGISTER
1321	002076	022777	177777	177034	CMP	#-1, #SWR	;;TRY TO REFERENCE HARDWARE SWR
1322	002104	001012			BNE	66\$	;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1323							;;AND THE HARDWARE SWR IS NOT = -1
1324	002106	000403			BR	65\$	;;BRANCH IF NO TIMEOUT
1325	002110	012716	002116		MOV	#65\$, (SP)	;;SET UP FOR TRAP RETURN
1326	002114	000002			RTI		
1327	002116	012737	000176	001140	MOV	#SWREG, SWR	;;POINT TO SOFTWARE SWR
1328	002124	012737	000174	001142	MOV	#DISPREG, DISPLAY	
1329	002132	012637	000004		MOV	(SP)+, #ERRVEC	;;RESTORE ERROR VECTOR
1330							
1331	002136	005227	177777		INC	#-1	;;FIRST START ?
1332	002142	001002			BNE	1\$	;;BR IF NOT
1333	002144	104401	016530		TYPE	TITLE	;;TYPE PROGRAM NAME
1334	002150	004737	015052		JSR	PC, STKINT	;;SETUP THE TTY KEYBOARD
1335					GET	VALUE FOR SOFTWARE SWITCH REGISTER	
1336	002154	005737	000042		TST	#42	;;ARE WE RUNNING UNDER XXDP/ACT?
1337	002160	001006			BNE	67\$	;;BRANCH IF YES
1338	002162	023727	001140	000176	CMP	SWR, #SWREG	;;SOFTWARE SWITCH REG SELECTED?
1339	002170	001005			BNE	68\$	;;BRANCH IF NO
1340	002172	14406			GTSWR		;;GET SOFT-SWR SETTINGS
1341	002174	000403			BR	68\$	
1342	002176	112737	000001	001134	MOV	#1, SAUTOB	;;SET AUTO-MODE INDICATOR
1343	002204						
1344	002204	004737	002576		JSR	PC, CHANGE	;;CHECK/CHANGE THE RH11 ADDRESS
1345	002210	104401	016625		TYPE	,ENTERA	;;ENTER DRIVE ADDRESS
1346	002214	104412			RDOCT		;;GET THE ADDRESS
1347	002216	012637	001224		MOV	(SP)+, PORTA	;;STORE THE ADDRESS
1348	002222	023727	001224	000007	CMP	PORTA, #7	;;SEE IF ADDRESS TOO LARGE
1349	002230	101403			BLOS	2\$	;;BR IF NOT
1350	002232	104401	016655		TYPE	ADRERR	;;TYPE ADDRESS ERROR MESSAGE
1351	002236	000744			BR	1\$	;;TRY AGAIN
1352	002240	013737	001224	001226	MOV	PORTA, PORTB	;;GENERATE THE PORT B ADDRESS
1353	002246	005237	001226		INC	PORTB	;;INCREMENT THE ADDRESS
1354	002252	042737	000016	001226	BIC	#16, PORTB	;;LEAVE BIT 0
1355	002260	013746	001224		MOV	PORTA, -(SP)	;;PUT PORT A ADDRESS ON THE STACK
1356	002264	042716	177771		BIC	#16, (SP)	;;SAVE BITS 1 & 2
1357	002270	052637	001226		BIS	(SP)+, PORTB	;;SET BITS 1 & 2 IN PORT B ADDRESS
1358	002274	104401	016677		TYPE	PORTA IS	;;PORT A ADDRESS IS
1359	002300	013746	001224		MOV	PORTA, -(SP)	;;SAVE PORTA FOR TYPEOUT
1360							;;TYPE PORT A ADDRESS
1361	002304	104403			TYPOS		;;GO TYPE--OCTAL ASCII
1362	002306	001			.BYTE	1	;;TYPE 1 DIGIT(S)
1363	002307	000			.BYTE	0	;;SUPPRESS LEADING ZEROS
1364	002310	104401	016725		TYPE	PORTB IS	;;PORT B ADDRESS IS
1365	002314	013746	001226		MOV	PORTB, -(SP)	;;SAVE PORTB FOR TYPEOUT
1366							;;TYPE PORT B ADDRESS
1367	002320	104403			TYPOS		;;GO TYPE--OCTAL ASCII
1368	002322	001			.BYTE	1	;;TYPE 1 DIGIT(S)
1369	002323	000			.BYTE	0	;;SUPPRESS LEADING ZEROS

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34

MACY11 30(1046) 21-NOV-77 14:15 PAGE 27
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0027

1370	002324	104401	001207		TYPE	SCRLF	4 OTHER CR-LF
1371	002330	013737	001224	001230	MOV	PORTA,PORTC	GENERATE ADDRESS OF DRIVE NOT TESTED
1372	002336	062737	000006	001230	ADD	#6,PORTC	COMPLEMENT SOME BITS
1373	002344	042737	177770	001230	BIC	#1C7,PORTC	SAVE ONLY LOWER BITS
1374	002352	013701	001224		MOV	PORTA,R1	USE PORT A ADDRESS AS INDEX
1375	002356	116137	024420	001232	MOVB	ATABIT(R1),ASR1	GET ATTENTION BIT FOR DRIVE
1376	002364	005037	001256		CLR	TIMEA	CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1377	002370	005037	001260		CLR	TIMEAP	CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1378	002374	005037	001262		CLR	TIMEB	CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1379	002400	005037	001264		CLR	TIMEBP	CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1380	002404	004737	013246		JSR	PC,CKCLK	SETUP CLOCK
1381	002410	000137	002424		JMP	EXEC	CLOCK HAS BEEN STARTED
1382	002414	104401	016753		TYPE	,NOCLOCK	NO CLOCK ON SYSTEM
1383	002420	000000		3S:	HALT		FATAL ERROR
1384	002422	000776			BR	3S	INTERLOCK THE HALT
1385							
1386							
1387							
1388	002424	000005			EXEC: RESET		CLEAR EVERYTHING
1389	002426	005037	177776		CLR	PS	CLEAR THE PROCESSOR STATUS WORD
1390	002432	104401	001207		TYPE	SCRLF	CR-LF
1391	002436	013700	001272		MOV	\$RMADR,RO	RH11 ADDRESS FOR INDEXING
1392	002442	012706	001100		MOV	#STACK,SP	LOAD STACK POINTER
1393	002446	004737	013246		JSR	PC,CKCLK	START THE CLOCK
1394	002452	000240			NOP		RETURN IF NO CLOCK
1395	002454	004737	015052		JSR	PC,STKINT	INITIALIZE THE KEYBOARD
1396	002460	005037	001256		CLR	KYBCTL	CLEAR SINGLE TEST INDICATOR
1397	002464	005037	001100		CLR	SPASS	CLEAR THE PASS COUNT
1398	002470	112737	000001	001115	MOVB	#1,SERMAX	SET ERROR MAX TO 1
1399	002476	012737	002476	001106	MOV	#,SLPADR	INITIAL SETTING FOR LOOP ADDRESS
1400	002504	012737	002504	001110	MOV	#,SLPERR	INITIAL SETTING FOR LOOP ON ERROR ADDRESS
1401	002512	104401	017022		1S: TYPE	,TESTNO	ASK FOR TEST NUMBER
1402	002516	104412			RDOCT		GET THE NUMBER
1403	002520	012601			MOV	(SP)+,R1	PUT ENTRY INTO R1
1404	002522	001002			BNE	2S	BR IF NOT ZERO
1405	002524	000137	002720		JMP	TST1	ENTER ZERO - PERFORM ALL TESTS
1406	002530	020137	024430		2S: CMP	R1,MAXTN	SEE IF NUMBER GREATER THAN MAXIMUM
1407	002534	003403			BLE	3S	BR IF LESS OR EQUAL
1408	002536	104401	017042		TYPE	BADNO	BAD ENTRY
1409	002542	000763			BR	1S	TRY AGAIN
1410	002544	005301		3S:	DEC	R1	DECREMENT ENTRY
1411	002546	006301			ASL	R1	SHIFT IT LEFT
1412	002550	016137	024400	002574	MOV	TSTADR(R1),4S	GET THE TEST ADDRESS
1413	002556	005237	001266		INC	KYBCTL	SET SINGLE TEST INDICATOR
1414	002562	012737	000001	001104	MOV	#1,SICNT	PRESET ITERATION COUNT
1415	002570	000177	000000		JMP	24S	GO TO THE SELECTED TEST
1416	002574	000000		4S:	.WORD	0	TEST ADDRESS GOES HERE
1417							
1418							
1419							
1420	002576	005737	001270		CHANGE: TST	CHGADR	CHANGE THE ADDRESS ?
1421	002602	001421			BEQ	3S	BR IF NOT
1422	002604	005037	001270		CLR	CHGADR	CLEAR THE INDICATOR
1423	002610	104401	017070		1S: TYPE	,ADDRIS	TYPE OUT WHAT THE PRESENT ADDRESS IS
1424	002614	013746	001272		MOV	\$RMADR,-(SP)	PUT THE ADDRESS ON THE STACK
1425	002620	104402			TYP0C		TYPE THE ACTUAL ADDRESS

```

1426 002622 104401 001207      TYPE      ,SCRLF      ;CR-LF
1427 002626 104401 017150      TYPE      ,NTRH11     ;ASK FOR NEW ADDRESS
1428 002632 104412              RDOCT              ;
1429 002634 005716              TST      (SP)        ;0 OR 'CR' ENTERED ?
1430 002636 001402              BEQ      2$          ;BR IF EITHER ENTERED (NO ADDRESS CHANGE)
1431 002640 011637 001272      MOV      (SP), $RMADR  ;NEW RH11 ADDRESS
1432 002644 005726              TST      (SP)+      ;CORRECT THE STACK POINTER
1433 002646 012737 002666 000004 2$:      MOV      #4$ ,2$4  ;LOAD TRAP ADDRESS
1434 002654 013700 001272      MOV      $RMADR, RO   ;RH11 ADDRESS
1435 002660 005760 000002      TST      RMWC(RO)    ;SEE IF RH11 RESPONDS AT THAT ADDRESS
1436 002664 000404              BR      5$          ;BR, RH11 ALIVE AT PRESENT ADDRESS
1437 002666 104025              4$:      ERROR 2$      ;NO RESPONSE TO ADDRESS
1438 002670 062706 000004      ADD      #4, SP      ;RESET THE STACK POINTER
1439 002674 000745              BR      1$          ;GET ADDRESS AGAIN
1440 002676 012737 000006 000004 5$:      MOV      #6, 2$4  ;RESTORE THE VECTOR
1441 002704 000207              RTS      PC          ;RETURN
1442
1443      ;;*****
1444      .SBTTL  *** TESTS ***
1445      ;;*****
1446
1447      TST1AA: MOV      $RMADR, RO   ;:RESTORE RO AFTER END OF PASS
1448      MOV      #CLR, RMCS2(RO) ;:CLEAR MASSBUS
1449
1450 002706 013700 001272 000010      ;;*****
1451 002712 012760 000040      *TEST 1      DRIVE ACCESS TEST
1452      ;;*****
1453      *
1454      *VERIFY THAT THE DRIVE CAN BE ACCESSED THROUGH BOTH PORTS
1455      *
1456      * A. SELECT DRIVE, VERIFY THAT THE DRIVE IS PRESENT, THAT THE
1457      * DRIVE IS A DUAL PORT RM03, THAT THE DRIVE IS ONLINE (RMDS1 HAS
1458      * 'MOL', 'PGM', 'DPR', & 'DRY' BITS SET), AND THE THE DRIVE SERIAL
1459      * NUMBER READ THROUGH BOTH PORTS IS THE SAME.
1460      *
1461      * B. THE TEST IS REPEATED THROUGH BOTH PORTS.
1462      *
1463      *
1464      *
1465      *
1466 002720      TST1:      TST      KYBCTL      ;PERFORMING ONLY SINGLE TESTS ?
1467 002720 005737 001266      BEQ      2$          ;BR IF NOT
1468 002724 001406              BPL      1$          ;BR IF JUST ENTERED TEST
1469 002726 100002              JMP      EXEC      ;RETURN & GET NEXT TEST NUMBER
1470 002730 000137 002424      1$:      MOV      #-1, KYBCTL ;SET SINGLE TEST INDICATOR
1471 002734 012737 177777 001266 2$:      MOVB   #1, $TSTNM ;TEST NUMBER
1472 002742 112737 000001 001102      MOV      $TEST1, $LPADR ;LOAD LOOP ON TEST ADDRESS
1473 002750 012737 002772 001106      MOV      $TEST1, $LPERR ;LOAD LOOP ON ERROR ADDRESS
1474 002756 012737 002772 001110      MOV      #1, $TIMES ;DO 1 ITERATION
1475 002764 012737 000001 001176      TEST1: MOV      $STACK, SP ;SETUP THE STACK POINTER
1476 002772 012706 001100
1477
1478      ;;*****
1479      ;VERIFY THAT DRIVE IS PRESENT THROUGH PORTS A & B
1480
1481 002776 113760 001224 000010      MOVB   PORTA, RMCS2(RO) ;SELECT PORT A

```


1482	003004	013737	001224	001234	MOV	PORTA,PTNBR	;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1483	003012	005760	000012		TST	RMDS1(R0)	;SEE IF DRIVE (PORT A) PRESENT
1484	003016	005037	001244		CLR	CKERR	;CLEAR THE 'CHECK ERROR' INDICATOR
1485	003022	016037	000010	001126	MOV	RMCS2(R0), \$BDDAT	;GET CONTENTS OF RMCS2
1486	003030	012737	000010	001122	MOV	\$RMCS2, \$B0ADR	;FORM REGISTER ADDRESS OF ERROR MESSAGE
1487	003036	060037	001122		ADD	R0, \$B0ADR	;ADD RH11 BASE ADDRESS
1488	003042	005037	001124		CLR	\$GDDAT	;WHAT REGISTER SHOULD BE
1489	003046	013737	001126	001164	MOV	\$BDDAT, \$TMP0	;MOVE REGISTER CONTENTS TO 'STMP0'
1490	003054	042737	167777	001164	BIC	\$1CNE0, \$TMP0	;SAVE SPECIFIED BITS
1491	003062	023737	001124	001164	CMP	\$GDDAT, \$TMP0	;COMPARE THE BITS
1492	003070	001414			BEQ	64\$	;BR IF OK
1493	003072	013737	001126	001174	MOV	\$BDDAT, \$TMP4	;COPY 'BAD DATA'
1494	003100	042737	010000	001174	BIC	\$NED, \$TMP4	;CLEAR THE MASKED BITS
1495	003106	053737	001174	001124	BIS	\$TMP4, \$GDDAT	;OR' WITH GOOD DATA FOR TYPEOUT
1496	003114	104001			ERROR	1	;TYPE MESSAGE 1
1497	003116	005137	001244		COM	CKERR	;SET THE REGISTER COMPARE ERROR INDICATOR
1498	003122	000240			NOP		
1499	003124	005737	001244		TST	CKERR	;WAS 'NED' SET ?
1500	003130	001403			BEQ	+.10	;BR IF NOT
1501	003132	012760	000040	000010	MOV	\$CLR, RMCS2(R0)	;ISSUE MASSBUS INIT TO CLEAR 'NED'
1502	003140	113760	001226	000010	MOVB	PORTB, RMCS2(R0)	;SELECT PORT B
1503	003146	013737	001226	001234	MOV	PORTB, PTNBR	;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1504	003154	005760	000012		TST	RMDS1(R0)	;SEE IF DRIVE (PORT B) PRESENT
1505	003160	005037	001244		CLR	CKERR	;CLEAR THE 'CHECK ERROR' INDICATOR
1506	003164	016037	000010	001126	MOV	RMCS2(R0), \$BDDAT	;GET CONTENTS OF RMCS2
1507	003172	012737	000010	001122	MOV	\$RMCS2, \$B0ADR	;FORM REGISTER ADDRESS OF ERROR MESSAGE
1508	003200	060037	001122		ADD	R0, \$B0ADR	;ADD RH11 BASE ADDRESS
1509	003204	005037	001124		CLR	\$GDDAT	;WHAT REGISTER SHOULD BE
1510	003210	013737	001126	001164	MOV	\$BDDAT, \$TMP0	;MOVE REGISTER CONTENTS TO 'STMP0'
1511	003216	042737	167777	001164	BIC	\$1CNE0, \$TMP0	;SAVE SPECIFIED BITS
1512	003224	023737	001124	001164	CMP	\$GDDAT, \$TMP0	;COMPARE THE BITS
1513	003232	001414			BEQ	66\$	;BR IF OK
1514	003234	013737	001126	001174	MOV	\$BDDAT, \$TMP4	;COPY 'BAD DATA'
1515	003242	042737	010000	001174	BIC	\$NED, \$TMP4	;CLEAR THE MASKED BITS
1516	003250	053737	001174	001124	BIS	\$TMP4, \$GDDAT	;OR' WITH GOOD DATA FOR TYPEOUT
1517	003256	104001			ERROR	1	;TYPE MESSAGE 1
1518	003260	005137	001244		COM	CKERR	;SET THE REGISTER COMPARE ERROR INDICATOR
1519	003264	000240			NOP		
1520	003266	005737	001244		TST	CKERR	;WAS 'NED' SET ?
1521	003272	001403			BEQ	+.10	;BR IF NOT
1522	003274	012760	000040	000010	MOV	\$CLR, RMCS2(R0)	;ISSUE MASSBUS INIT TO CLEAR 'NED'
1523							
1524							
1525							
1526							
1527	003302	113760	001224	000010	MOVB	PORTA, RMCS2(R0)	;SELECT PORT A
1528	003310	013737	001224	001234	MOV	PORTA, PTNBR	;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1529	003316	005037	001244		CLR	CKERR	;CLEAR THE 'CHECK ERROR' INDICATOR
1530	003322	016037	000026	001126	MOV	RMDT(R0), \$BDDAT	;GET CONTENTS OF RMDT
1531	003330	012737	000026	001122	MOV	\$RMDT, \$B0ADR	;FORM REGISTER ADDRESS OF ERROR MESSAGE
1532	003336	060037	001122		ADD	R0, \$B0ADR	;ADD RH11 BASE ADDRESS
1533	003342	012737	024024	001124	MOV	\$024024, \$GDDAT	;WHAT REGISTER SHOULD BE
1534	003350	022737	024025	001126	CMP	\$24025, \$BDDAT	;DUAL PORT RM02 DRIVE ?
1535	003356	001426			BEQ	68\$	;BRANCH IF SO
1536	003360	013737	001126	001164	MOV	\$BDDAT, \$TMP0	;MOVE REGISTER CONTENTS TO 'STMP0'
1537	003366	042737	000003	001164	BIC	\$1C17774, \$TMP0	;SAVE SPECIFIED BITS

;CONFIRM THAT DRIVE IS AN RM03 AND IS DUAL PORT


```

1538 003374 023737 001124 001164      CMP      $GDDAT,$STMP0      ;COMPARE THE BITS
1539 003402 001414                      BEQ      68$                ;BR IF OK
1540 003404 013737 001126 001174      MOV      $BDDAT,$STMP4      ;COPY 'BAD DATA'
1541 003412 042737 177774 001174      BIC      #177774,$STMP4      ;CLEAR THE MASKED BITS
1542 003420 053737 001174 001124      BIS      $STMP4,$GDDAT      ;'OR' WITH GOOD DATA FOR TYPEOUT
1543 003426 104002                      ERROR    2                ;TYPE MESSAGE 2
1544 003430 005137 001244                      COM      CKERR          ;SET THE REGISTER COMPARE ERROR INDICATOR
1545 003434 000240                      NOP
1546 003436 113760 001226 000010      MOV      PORTB,RMCS2(R0)    ;SELECT PORT B
1547 003444 013737 001226 001234      MOV      PORTB,PTNBR        ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1548 003452 005037 001244                      CLR      CKERR              ;CLEAR THE 'CHECK ERROR' INDICATOR
1549 003456 016037 000026 001126      MOV      RMDT(R0),$BDDAT     ;GET CONTENTS OF RMDT
1550 003464 012737 000026 001122      MOV      #RMDT,$BDAOR        ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1551 003472 060037 001122                      ADD      R0,$BDAOR           ;ADD RH11 BASE ADDRESS
1552 003476 012737 024024 001124      MOV      #024024,$GDDAT      ;WHAT REGISTER SHOULD BE
1553 003504 022737 024025 001126      CMP      #24025,$BDDAT       ;DUAL PORT RM02 DRIVE ?
1554 003512 001426                      BEQ      70$                ;BRANCH IF SO
1555 003514 013737 001126 001164      MOV      $BDDAT,$STMP0      ;MOVE REGISTER CONTENTS TO 'STMP0'
1556 003522 042737 000003 001164      BIC      #1C177774,$STMP0    ;SAVE SPECIFIED BITS
1557 003530 023737 001124 001164      CMP      $GDDAT,$STMP0      ;COMPARE THE BITS
1558 003536 001414                      BEQ      70$                ;BR IF OK
1559 003540 013737 001126 001174      MOV      $BDDAT,$STMP4      ;COPY 'BAD DATA'
1560 003546 042737 177774 001174      BIC      #177774,$STMP4      ;CLEAR THE MASKED BITS
1561 003554 053737 001174 001124      BIS      $STMP4,$GDDAT      ;'OR' WITH GOOD DATA FOR TYPEOUT
1562 003562 104002                      ERROR    2                ;TYPE MESSAGE 2
1563 003564 005137 001244                      COM      CKERR          ;SET THE REGISTER COMPARE ERROR INDICATOR
1564 003570 000240                      NOP
1565
1566      ;*****
1567      ;VERIFY THROUGH BOTH PORTS THAT THE DRIVE IS ON LINE AND IN NEUTRAL
1568
1569 003572 113760 001224 000010      MOV      PORTA,RMCS2(R0)    ;SELECT PORT A
1570 003600 013737 001224 001234      MOV      PORTA,PTNBR        ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1571 003606 005037 001244                      CLR      CKERR              ;CLEAR THE 'CHECK ERROR' INDICATOR
1572 003612 016037 000012 001126      MOV      RMD1(R0),$BDDAT     ;GET CONTENTS OF RMD1
1573 003620 012737 000012 001122      MOV      #RMD1,$BDAOR        ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1574 003626 060037 001122                      ADD      R0,$BDAOR           ;ADD RH11 BASE ADDRESS
1575 003632 012737 001000 001124      MOV      #PGM,$GDDAT         ;WHAT REGISTER SHOULD BE
1576 003640 013737 001126 001164      MOV      $BDDAT,$STMP0      ;MOVE REGISTER CONTENTS TO 'STMP0'
1577 003646 042737 176777 001164      BIC      #1CPGM,$STMP0       ;SAVE SPECIFIED BITS
1578 003654 023737 001124 001164      CMP      $GDDAT,$STMP0      ;COMPARE THE BITS
1579 003662 001414                      BEQ      72$                ;BR IF OK
1580 003664 013737 001126 001174      MOV      $BDDAT,$STMP4      ;COPY 'BAD DATA'
1581 003672 042737 001000 001174      BIC      #PGM,$STMP4         ;CLEAR THE MASKED BITS
1582 003700 053737 001174 001124      BIS      $STMP4,$GDDAT      ;'OR' WITH GOOD DATA FOR TYPEOUT
1583 003706 104003                      ERROR    3                ;TYPE MESSAGE 3
1584 003710 005137 001244                      COM      CKERR          ;SET THE REGISTER COMPARE ERROR INDICATOR
1585 003714 000240                      NOP
1586 003716 005037 001244                      CLR      CKERR              ;CLEAR THE 'CHECK ERROR' INDICATOR
1587 003722 016037 000012 001126      MOV      RMD1(R0),$BDDAT     ;GET CONTENTS OF RMD1
1588 003730 012737 000012 001122      MOV      #RMD1,$BDAOR        ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1589 003736 060037 001122                      ADD      R0,$BDAOR           ;ADD RH11 BASE ADDRESS
1590 003742 012737 010600 001124      MOV      #MOL:DPR:DRY,$GDDAT ;WHAT REGISTER SHOULD BE
1591 003750 013737 001126 001164      MOV      $BDDAT,$STMP0      ;MOVE REGISTER CONTENTS TO 'STMP0'
1592 003756 042737 167177 001164      BIC      #1C10600,$STMP0     ;SAVE SPECIFIED BITS
1593 003764 023737 001124 001164      CMP      $GDDAT,$STMP0      ;COMPARE THE BITS

```



```

1594 003772 001414      BEQ      74$      ;BR IF OK
1595 003774 013737 001126 001174      MOV      $BDDAT,$TMP4 ;COPY 'BAD DATA'
1596 004002 042737 010600 001174      BIC      #10600,$TMP4 ;CLEAR THE MASKED BITS
1597 004010 053737 001174 001124      BIS      $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
1598 004016 104004      ERROR    4      ;TYPE MESSAGE 4
1599 004020 005137 001244      COM      CKERR      ;SET THE REGISTER COMPARE ERROR INDICATOR
1600 004024 000240      NOP
1601 004026 113760 001226 000010 74$: MOV     PORTB,RMCS2(R0) ;SELECT PORT B
1602 004034 013737 001226 001234      MOV     PORTB,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1603 004042 005037 001244      CLR      CKERR      ;CLEAR THE 'CHECK ERROR' INDICATOR
1604 004046 016037 000012 001126      MOV     RMDS1(R0),$BDDAT ;GET CONTENTS OF RMDS1
1605 004054 012737 000012 001122      MOV     #RMDS1,$BDDADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1606 004062 060037 001122      ADD     R0,$BDDADR ;ADD RM11 BASE ADDRESS
1607 004066 012737 001000 001124      MOV     #PGM,$GDDAT ;WHAT REGISTER SHOULD BE
1608 004074 013737 001126 001164      MOV     $BDDAT,$TMP0 ;MOVE REGISTER CONTENTS TO '$TMP0'
1609 004102 042737 176777 001164      BIC     #1CPGM,$TMP0 ;SAVE SPECIFIED BITS
1610 004110 023737 001124 001164      CMP     $GDDAT,$TMP0 ;COMPARE THE BITS
1611 004116 001414      BEQ      76$      ;BR IF OK
1612 004120 013737 001126 001174      MOV     $BDDAT,$TMP4 ;COPY 'BAD DATA'
1613 004126 042737 001000 001174      BIC     #PGM,$TMP4 ;CLEAR THE MASKED BITS
1614 004134 053737 001174 001124      BIS     $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
1615 004142 104003      ERROR    3      ;TYPE MESSAGE 3
1616 004144 005137 001244      COM      CKERR      ;SET THE REGISTER COMPARE ERROR INDICATOR
1617 004150 000240      NOP
1618 004152 005037 001244      CLR      CKERR      ;CLEAR THE 'CHECK ERROR' INDICATOR
1619 004156 016037 000012 001126      MOV     RMDS1(R0),$BDDAT ;GET CONTENTS OF RMDS1
1620 004164 012737 000012 001122      MOV     #RMDS1,$BDDADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1621 004172 060037 001122      ADD     R0,$BDDADR ;ADD RM11 BASE ADDRESS
1622 004176 012737 010600 001124      MOV     #MOL:DPR:DRY,$GDDAT ;WHAT REGISTER SHOULD BE
1623 004204 013737 001126 001164      MOV     $BDDAT,$TMP0 ;MOVE REGISTER CONTENTS TO '$TMP0'
1624 004212 042737 167177 001164      BIC     #1C10600,$TMP0 ;SAVE SPECIFIED BITS
1625 004220 023737 001124 001164      CMP     $GDDAT,$TMP0 ;COMPARE THE BITS
1626 004226 001414      BEQ      78$      ;BR IF OK
1627 004230 013737 001126 001174      MOV     $BDDAT,$TMP4 ;COPY 'BAD DATA'
1628 004236 042737 010600 001174      BIC     #10600,$TMP4 ;CLEAR THE MASKED BITS
1629 004244 053737 001174 001124      BIS     $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
1630 004252 104004      ERROR    4      ;TYPE MESSAGE 4
1631 004254 005137 001244      COM      CKERR      ;SET THE REGISTER COMPARE ERROR INDICATOR
1632 004260 000240      NOP
1633
1634 ;*****
1635 ;VERIFY THAT DRIVE SERIAL NUMBER SEEN THROUGH BOTH PORTS IS THE SAME
1636
1637 004262 113760 001224 000010      MOV     PORTA,RMCS2(R0) ;SELECT PORT A
1638 004270 016037 000030 001124      MOV     RMDSN(R0),$GDDAT ;STORE THE PORT A SERIAL NUMBER
1639 004276 113760 001226 000010      MOV     PORTB,RMCS2(R0) ;SELECT PORT B
1640 004304 016037 000030 001126      MOV     RMDSN(R0),$BDDAT ;STORE THE PORT B SERIAL NUMBER
1641 004312 023737 001124 001126      CMP     $GDDAT,$BDDAT ;ARE THEY THE SAME ?
1642 004320 001406      BEQ      1$      ;BR IF THEY ARE
1643 004322 104005      ERROR    5      ;REPORT THE ERROR
1644 004324 032777 100000 174606      BIT     #SW15,$SWR ;HALT ON ERROR ?
1645 004332 001001      BNE     1$      ;BR IF SET - PROGRAM HAS ALREADY HALTED
1646 004334 000000      HALT
1647 004336 000004      SCOPE
1648
1649

```

1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705

004340	005737	001266	
004340	001406		
004346	100002		
004350	000137	002424	
004354	012737	177777	001266
004362	112737	000002	001102
004370	012737	004412	001106
004376	012737	004412	001110
004404	012737	000001	001176
004412	012706	001100	
004416	113760	001224	000010
004424	013737	001224	001234
004432	012760	000011	000000
004440	012760	000021	000000
004446	012760	010000	000032
004454	005037	001244	
004460	016037	000012	001126
004466	012737	000012	001122
004474	060037	001122	
004500	012737	011700	001124
004506	013737	001126	001164
004514	042737	106077	001164
004522	023737	001124	001164
004530	001414		
004532	013737	001126	001174
004540	042737	071700	001174
004546	053737	001174	001124
004554	104010		
004556	005137	001244	
004562	000240		

```

*****
: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
:        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$:   MOV      #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
2$:   MOV      #2,$TSTNM       ;TEST NUMBER
      MOV      #TEST2,$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
      MOV      PORTA,$MCS2($R0) ;SELECT PORT A
      MOV      PORTA,$PTNBR     ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT

```

```

*****
:SET VOLUME VALUE FOR PORT
*****

```

```

      MOV      #11,$MCS1($R0)  ;ISSUE A DRIVE CLEAR
      MOV      #21,$MCS1($R0)  ;ISSUE A READIN PRESET
      MOV      #FMT16,$RMOF($R0) ;SET FMT16

```

```

*****
:VERIFY THAT THE DRIVE STATUS IS CORRECT
*****

```

```

      CLR      CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
      MOV      $RMS1($R0),$SDDAT ;GET CONTENTS OF $RMS1
      MOV      $RMS1,$SBOADR    ;FORM REGISTER ADDRESS OF ERROR MESSAGE
      ADD      $R0,$SBOADR       ;ADD RH11 BASE ADDRESS
      MOV      #MOL!PGM!DPR!DRY!VV,$SDDAT ;WHAT REGISTER SHOULD BE
      MOV      $SDDAT,$TMP0      ;MOVE REGISTER CONTENTS TO '$TMP0'
      BIC      #C71700,$TMP0     ;SAVE SPECIFIED BITS
      CMP      $SDDAT,$TMP0      ;COMPARE THE BITS
      BEQ      64$              ;BR IF OK
      MOV      $SDDAT,$TMP4      ;COPY 'BAD DATA'
      BIC      #71700,$TMP4      ;CLEAR THE MASKED BITS
      BIS      $TMP4,$SDDAT      ;'OR' WITH GOOD DATA FOR TYPEOUT
      ERROR    10               ;TYPE MESSAGE 10
      COM      CKERR            ;SET THE REGISTER COMPARE ERROR INDICATOR
64$:   NOP

```


H03

CZRMHBO RM03/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 33
T2 SET 'VV' FOR PORT A

SEQ 0033

```

1706
1707
1708
1709
1710 004564 113760 001224 000010
1711 004572 013737 001224 001234
1712 004600 012760 000013 000000
1713
1714
1715
1716 004606 005037 001250
1717 004612 012737 000012 001122
1718 004620 060037 001122
1719 004624 012737 011600 001124
1720 004632 113760 001224 000010
1721 004640 016037 000012 001170
1722 004646 013737 001170 001164
1723 004654 042737 100100 001164
1724 004662 113760 001226 000010
1725 004670 016037 000012 001172
1726 004676 013737 001172 001166
1727 004704 042737 100100 001166
1728 004712 023737 001164 001166
1729 004720 001006
1730 004722 005737 001164
1731 004726 001037
1732 004730 104034
1733 004732 000137 005116
1734 004736 013737 001170 001126 66$:
1735 004744 013737 001226 001234
1736 004752 113760 001226 000010
1737 004760 005737 001164
1738 004764 001414
1739 004766 013737 001224 001234
1740 004774 013737 001172 001126
1741 005002 113760 001224 000010
1742 005010 005737 001166
1743 005014 001004
1744 005016 012737 177777 001250 67$:
1745 005024 104036
1746 005026 013737 001170 001126 68$:
1747 005034 013737 001224 001234
1748 005042 042737 100100 001170
1749 005050 023737 001124 001170
1750 005056 001401
1751 005060 104037
1752 005062 013737 001172 001126 69$:
1753 005070 013737 001226 001234
1754 005076 042737 100100 001172
1755 005104 023737 001124 001172
1756 005112 001401
1757 005114 104037
1758 005116 000240
1759 005120 000004
1760
1761

```

```

;*****
;RELEASE THE DRIVE FROM PORT A
MOV      PORTA, RMCS2(R0) ;SELECT PORT A
MOV      PORTA, PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
MOV      #13, RMCS1(R0) ;ISSUE RELEASE THROUGH PORT A
;VERIFY THAT THE DRIVE IS IN NEUTRAL
CLR      RELERR ;CLEAR THE 'RELEASE ERROR' INDICATOR
MOV      #RMS1, $BDDAT ;FORM THE ADDRESS OF RMS1 FOR TYPEOUT
ADD      R0, $BDDAT ;ADD THE I/O BASE ADDRESS
MOV      #MOL:PGM:OPR:DRY, $GDDAT ;COMPARISON CONSTANT
MOV      PORTA, RMCS2(R0) ;SELECT PORT A.
MOV      RMS1(R0), STMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
MOV      STMP2, STMP0 ;COPY IT INTO 'STMP0'
BIC      #ATA:VV, STMP0 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
MOV      PORTB, RMCS2(R0) ;SELECT PORT B.
MOV      RMS1(R0), STMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
MOV      STMP3, STMP1 ;COPY IT INTO 'STMP1'
BIC      #ATA:VV, STMP1 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
CMP      STMP0, STMP1 ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
BNE      66$ ;BR IF NOT
TST      STMP0 ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
BNE      68$ ;BR IF NOT
ERROR    34 ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
JMP      70$ ;BYPASS THE REST OF THE CHECKS
MOV      STMP2, $BDDAT ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
MOV      PORTB, PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
MOV      PORTB, RMCS2(R0) ;SELECT PORT B.
TST      STMP0 ;SEE IF STATUS EQ 0 FROM PORT A.
BEQ      67$ ;BR IF ZERO
MOV      PORTA, PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
MOV      STMP3, $BDDAT ;'BAD DATA' FOR ERROR TYPE OUT
MOV      PORTA, RMCS2(R0) ;SELECT PORT A.
TST      STMP1 ;SEE IF STATUS EQ ZERO FROM PORT B.
BNE      68$ ;BR IF NOT
MOV      #1, RELERR ;SET 'RELEASE ERROR' INDICATOR
ERROR    36 ;TYPE ERROR MESSAGE 36
MOV      STMP2, $BDDAT ;LOOK FOR BIT FAILURES WHEN RMS1 READ
MOV      PORTA, PTNBR ;CHANGE PORT NUMBER
BIC      #ATA:VV, STMP2 ;DON'T CHECK ATTN BIT OR VV BIT
CMP      $GDDAT, STMP2 ;ALL BITS OK ?
BEQ      69$ ;BR IF OK FROM PORT A.
ERROR    37 ;REPORT ERROR
MOV      STMP3, $BDDAT ;CHECK RMS1 FOR BIT FAILURES - FROM PORT B.
MOV      PORTB, PTNBR ;CHANGE PORT NUMBER
BIC      #ATA:VV, STMP3 ;DON'T CHECK ATTN BIT OR VV BIT
CMP      $GDDAT, STMP3 ;SEE IF READ OK FROM PORT B.
BEQ      70$ ;BR IF OK
ERROR    37 ;REPORT THE ERROR
NOP
SCOPE
;LOOP ?
;*****

```

```

1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776 005122
1777 005122 005737 001266
1778 005126 001406
1779 005130 100002
1780 005132 000137 002424
1781 005136 012737 177777 001266
1782 005144 112737 000003 001102
1783 005152 012737 005174 001106
1784 005160 012737 005174 001110
1785 005166 012737 000001 001176
1786 005174 012706 001100
1787 005200 113760 001226 000010
1788 005206 013737 001226 001234
1789
1790
1791
1792
1793 005214 012760 000011 000000
1794 005222 012760 000021 000000
1795 005230 012760 010000 000032
1796
1797
1798
1799
1800 005236 005037 001244
1801 005242 016037 000012 001126
1802 005250 012737 000012 001122
1803 005256 060037 001122
1804 005262 012737 011700 001124
1805 005270 013737 001126 001164
1806 005276 042737 106077 001164
1807 005304 023737 001124 001164
1808 005312 001414
1809 005314 013737 001126 001174
1810 005322 042737 071700 001174
1811 005330 053737 001174 001124
1812 005336 104010
1813 005340 005137 001244
1814 005344 000240
1815
1816
1817

; *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.
; *
; *****
; 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: MOV #3, STSTNM ; TEST NUMBER
; MOV #TEST3, SLPADR ; LOAD LOOP ON TEST ADDRESS
; MOV #TEST3, SLPERR ; LOAD LOOP ON ERROR ADDRESS
; MOV #1, STTIMES ; DO 1 ITERATION
TEST3: MOV #STACK, SP ; SETUP THE STACK POINTER
; MOV #PORTB, RMCS2(R0) ; SELECT PORT B
; MOV #PORTB, PTNBR ; MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT

; *****
; SET VOLUME VALUE FOR PORT
;
; MOV #11, RMCS1(R0) ; ISSUE A DRIVE CLEAR
; MOV #21, RMCS1(R0) ; ISSUE A READIN PRESET
; MOV #FMT16, RMOF(R0) ; SET FMT16

; *****
; VERIFY THAT THE DRIVE STATUS IS CORRECT
;
; CLR CKERR ; CLEAR THE 'CHECK ERROR' INDICATOR
; MOV RMDS1(R0), $BDDAT ; GET CONTENTS OF RMDS1
; MOV #RMDS1, $B0ADR ; FORM REGISTER ADDRESS OF ERROR MESSAGE
; ADD R0, $B0ADR ; ADD RH11 BASE ADDRESS
; MOV #MOL!PGM!DPR!DRY!VV, $GDDAT ; WHAT REGISTER SHOULD BE
; MOV $BDDAT, $TMP0 ; MOVE REGISTER CONTENTS TO '$TMP0'
; BIC #1C71700, $TMP0 ; SAVE SPECIFIED BITS
; CMP $GDDAT, $TMP0 ; COMPARE THE BITS
; BEQ 64$ ; BR IF OK
; MOV $BDDAT, $TMP4 ; COPY 'BAD DATA'
; BIC #71700, $TMP4 ; CLEAR THE MASKED BITS
; BIS $TMP4, $GDDAT ; 'OR' WITH GOOD DATA FOR TYPEOUT
; ERROR 10 ; TYPE MESSAGE 10
; COM CKERR ; SET THE REGISTER COMPARE ERROR INDICATOR
; JNP 64$
; *****

```


Address	Hex	Hex	Hex	Hex	Label	Assembly	Comment
1818						;RELEASE THE DRIVE FROM PORT B	
1819							
1820	005346	113760	001226	000010		MOVB PORTB, RMCS2(R0)	:SELECT PORT B
1821	005354	013737	001226	001234		MOV PORTB, PTNBR	:MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1822	005362	012760	000013	000000		MOV #13, RMCS1(R0)	:ISSUE RELEASE THROUGH PORT B
1823							
1824							
1825						;VERIFY THAT THE DRIVE IS IN NEUTRAL	
1826	005370	005037	001250			CLR RELERR	:CLEAR THE 'RELEASE ERROR' INDICATOR
1827	005374	012737	000012	001122		MOV #RMD51, SBDAOR	:FORM THE ADDRESS OF RMD51 FOR TYPEOUT
1828	005402	060037	001122			ADD R0, SBDAOR	:ADD THE I/O BASE ADDRESS
1829	005406	012737	011600	001124		MOV #MOL!PGM!DPR!DRY, SGDDAT	:COMPARISON CONSTANT
1830	005414	113760	001224	000010		MOVB PORTA, RMCS2(R0)	:SELECT PORT A.
1831	005422	016037	000012	001170		MOV RMD51(R0), STMP2	:GET THE DRIVE STATUS REGISTER FROM PORT A.
1832	005430	013737	001170	001164		MUV STMP2, STMP0	:COPY IT INTO 'STMP0'
1833	005436	042737	100100	001164		BIC #ATA!VV, STMP0	:CLEAR PORT DEPENDENT BITS FROM THE COPY
1834	005444	113760	001226	000010		MOVB PORTB, RMCS2(R0)	:SELECT PORT B.
1835	005452	016037	000012	001172		MOV RMD51(R0), STMP3	:GET THE DRIVE STATUS REGISTER FROM PORT B.
1836	005460	013737	001172	001166		MOV STMP3, STMP1	:COPY IT INTO 'STMP1'
1837	005466	042737	100100	001166		BIC #ATA!VV, STMP1	:CLEAR PORT DEPENDENT BITS FROM THE COPY
1838	005474	023737	001164	001166		CMP STMP0, STMP1	:IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
1839	005502	001006				BNE 66\$	:BR IF NOT
1840	005504	005737	001164			TST STMP0	:REGISTERS ARE THE SAME: ARE THEY ZERO ?
1841	005510	001037				BNE 68\$	:BR IF NOT
1842	005512	104034				ERROR 34	:REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
1843	005514	000137	005700			JMP 70\$	:BYPASS THE REST OF THE CHECKS
1844	005520	013737	001170	001126	66\$:	MOV STMP2, SBDDAT	:SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
1845	005526	013737	001226	001234		MOV PORTB, PTNBR	:SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
1846	005534	113760	001226	000010		MOVB PORTB, RMCS2(R0)	:SELECT PORT B.
1847	005542	005737	001164			TST STMP0	:SEE IF STATUS EQ 0 FROM PORT A.
1848	005546	001414				BEQ 67\$	:BR IF ZERO
1849	005550	013737	001224	001234		MOV PORTA, PTNBR	:SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
1850	005556	013737	001172	001126		MOV STMP3, SBDDAT	: 'BAD DATA' FOR ERROR TYPE OUT
1851	005564	113760	001224	000010		MOVB PORTA, RMCS2(R0)	:SELECT PORT A.
1852	005572	005737	001166			TST STMP1	:SEE IF STATUS EQ ZERO FROM PORT B.
1853	005576	001004				BNE 68\$	:BR IF NOT
1854	005600	012737	177777	001250	67\$:	MOV # -1, RELERR	:SET 'RELEASE ERROR' INDICATOR
1855	005606	104036				ERROR 36	:TYPE ERROR MESSAGE 36
1856	005610	013737	001170	001126	68\$:	MOV STMP2, SBDDAT	:LOOK FOR BIT FAILURES WHEN RMD51 READ
1857	005616	013737	001224	001234		MOV PORTA, PTNBR	:CHANGE PORT NUMBER
1858	005624	042737	100100	001170		BIC #ATA!VV, STMP2	:DON'T CHECK ATTN BIT OR VV BIT
1859	005632	023737	001124	001170		CMP SGDDAT, STMP2	:ALL BITS OK ?
1860	005640	001401				BEQ 69\$	:BR IF OK FROM PORT A.
1861	005642	104037				ERROR 37	:REPORT ERROR
1862	005644	013737	001172	001126	69\$:	MOV STMP3, SBDDAT	:CHECK RMD51 FOR BIT FAILURES - FROM PORT B.
1863							

K03

CZRMHBO RMO3/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 36
T4 MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT A

SEQ 0036

```

1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889 005704
1890 005704 005737 001266
1891 005710 001406
1892 005712 100002
1893 005714 000137 002424
1894 005720 012737 177777 001266
1895 005726 112737 000004 001102
1896 005734 012737 005756 001106
1897 005742 012737 005756 001110
1898 005750 012737 000001 001176
1899 005756 012706 001100
1900 005762 005037 001256
1901 005766 005037 001260
1902
1903
1904
1905
1906 005772 005037 001252
1907 005776 012737 003720 001254
1908
1909
1910
1911
1912
1913 006004 113760 001224 000010
1914 006012 013737 001224 001236
1915 006020 005060 000012
1916 006024 113760 001226 000010
1917 006032 013737 001226 001234
1918 006040 013737 001226 001240
1919 006046 016037 000012 001126
1920 006054 010037 001122
1921 006060 062737 000012 001122
1922 006066 005037 001124
1923 006072 023737 001124 001126
1924 006100 001403
1925 006102 104030
1926 006104 000137 006620
1927 006110
1928 006110 113760 001224 000010
1929 006116 013737 001224 001234

*****
TEST 4 MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT A
*****
MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT A
*****
A. WRITE 0'S INTO RMD51 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, STSTNM ;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
MOVB PORTA, RMCS2(R0) ;SELECT PORT A
MOV PORTA, SEIZPT ;STORE SEIZING PORT'S ADDRESS
CLR RMD51(R0) ;WRITE RMD51
MOVB PORTB, RMCS2(R0) ;SELECT PORT B
MOV PORTB, PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
MOV PORTB, OPPRT ;'OPPOSITE' PORT ADDRESS
MOV RMD51(R0), $BDDAT ;SEE IF DRIVE SEIZED BY PORT A
MOV R0, $BDADR ;R011 BASE ADDRESS
ADD #RMD51, $BDADR ;GENERATE BAD REGISTER ADDRESS
CLR $GDDAT ;REGISTER SHOULD BE ZERO
CMP $GDDAT, $BDDAT ;IS THE REGISTER ZERO
BEQ 645 ;BR IF IT IS
ERROR 30 ;REPORT THE ERROR
JMP 45 ;BYPASS REST OF THE SUBTEST
645:
MOVB PORTA, RMCS2(R0) ;SELECT PORT A
MOV PORTA, PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT

```


L03

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046)
T421-NOV-77 14:15 PAGE 37
MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT A

SEQ 0037

```

1930 006124 016037 000012 001126 MOV RMDS1(R0), $BDDAT ;SEE IF SEIZING PORT SEES CORRECT STATUS
1931 006132 012737 011700 001124 MOV $MOL!PGM!DPR!DRY!VV, $GDDAT ;EXPECTED STATUS
1932 006140 013737 001124 001166 MOV $GDDAT, STMP1 ;USE GOOD DATA AS A MASK
1933 006146 005137 001166 COM STMP1 ;COMPLEMENT THE EXPECTED STATUS
1934 006152 013737 001126 001164 MOV $BDDAT, STMP0 ;SAVE THE ACTUAL STATUS
1935 006160 043737 001166 001164 BIC STMP1, STMP0 ;CLEAR UNWANTED BITS
1936 006166 023737 001124 001164 CMP $GDDAT, STMP0 ;ARE THE EXPECTED STATUS BITS SET ?
1937 006174 001401 BEQ 65$ ;BR IF THEY ARE
1938 006176 104031 ERROR 31 ;REPORT THE ERROR
1939 006200 000240 65$: NOP

;*****
;WAIT FOR PORT A TO TIMEOUT

1944 006202 113760 001226 000010 MOVB PORTB, RMCS2(R0) ;SELECT PORT B
1945 006210 013737 001226 001234 MOV PORTB, PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1946 006216 005760 000012 1$: TST RMDS1(R0) ;WAIT FOR THE DRIVE TO TIMEOUT
1947 006222 001006 BNE 2$ ;BR WHEN TIMEOUT OCCURS
1948 006224 005737 001254 TST WATCH ;CHECK WATCH
1949 006230 001372 BNE 1$ ;BR IF NOT ZERO
1950 006232 104006 ERROR 6 ;NO TIMEOUT WITHIN 2 SECONDS
1951 006234 000137 006272 JMP 3$ ;BYPASS THE REST OF THE TEST
1952 006240 013737 001252 001256 2$: MOV TIME, TIMEA ;SAVE THE ELAPSED TIME FOR PORT A
1953 006246 004537 013432 JSR RS, TOLER ;CALCULATE THE TOLERANCE
1954 006252 001256 .WORD TIMEA ;TIMEOUT VALUE FOR PORT A
1955 006254 012637 001260 MOV (SP)+, TIMEAP ;+25% TOLERANCE

;*****
;VERIFY THAT THE TIMEOUT ONE-SHOT VALUE IS AT LEAST 500 MS

1959 006260 023727 001256 000764 CMP TIMEA, #500. ;IS TIMEOUT VALUE AT LEAST 500 MS ?
1961 006266 103001 BHS 3$ ;BR IF IT IS
1962 006270 104007 ERROR 7 ;TIMEOUT LESS THAN 500 MS

;*****
;VERIFY THAT THE DRIVE RETURNED TO NEUTRAL AFTER PORT A TIMED OUT

1967 006272 3$:

;VERIFY THAT THE DRIVE IS IN NEUTRAL

1971 006272 005037 001250 CLR RELERR ;CLEAR THE 'RELEASE ERROR' INDICATOR
1972 006276 012737 000012 001122 MOV $RMDS1, $BDDADR ;FORM THE ADDRESS OF RMDS1 FOR TYPEOUT
1973 006304 060037 001122 ADD R0, $BDDADR ;ADD THE I/O BASE ADDRESS
1974 006310 012737 011700 001124 MOV $MOL!PGM!DPR!DRY!VV, $GDDAT ;COMPARISON CONSTANT
1975 006316 113760 001224 000010 MOVB PORTA, RMCS2(R0) ;SELECT PORT A.
1976 006324 016037 000012 001170 MOV RMDS1(R0), STMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
1977 006332 013737 001170 001164 MOV STMP2, STMP0 ;COPY IT INTO 'STMP0'
1978 006340 042737 100100 001164 BIC $ATA!VV, STMP0 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
1979 006346 113760 001226 000010 MOVB PORTB, RMCS2(R0) ;SELECT PORT B.
1980 006354 016037 000012 001172 MOV RMDS1(R0), STMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
1981 006362 013737 001172 001166 MOV STMP3, STMP1 ;COPY IT INTO 'STMP1'
1982 006370 042737 100100 001166 BIC $ATA!VV, STMP1 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
1983 006376 023737 001164 001166 CMP STMP0, STMP1 ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
1984 006404 001006 BNE 66$ ;BR IF NOT
1985 006406 005737 001164 TST STMP0 ;REGISTERS ARE THE SAME: ARE THEY ZERO ?

```


M03

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046)
T421-NOV-77 14:15 PAGE 38
MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT A

SEQ 0038

1986	006412	001045				BNE	68\$		BR IF NOT
1987	006414	104034				ERROR	34		REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
1988	006416	000137	006616			JMP	70\$		BYPASS THE REST OF THE CHECKS
1989	006422	013737	001170	001126	66\$:	MOV	STMP2, \$BDDAT		SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
1990	006430	013737	001226	001234		MOV	PORTB, PTNBR		SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
1991	006436	113760	001226	000010		MOVB	PORTB, RMCS2(R0)		SELECT PORT B.
1992	006444	005737	001164			TST	STMP0		SEE IF STATUS EQ 0 FROM PORT A.
1993	006450	001414				BEQ	67\$		BR IF ZERO
1994	006452	013737	001224	001234		MOV	PORTA, PTNBR		SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
1995	006460	013737	001172	001126		MOV	STMP3, \$BDDAT		'BAD DATA' FOR ERROR TYPE OUT
1996	006466	113760	001224	000010		MOVB	PORTA, RMCS2(R0)		SELECT PORT A.
1997	006474	005737	001166			TST	STMP1		SEE IF STATUS EQ ZERO FROM PORT B.
1998	006500	001012				BNE	68\$		BR IF NOT
1999	006502	012737	177777	001250	67\$:	MOV	#-1, RELERR		SET 'RELEASE ERROR' INDICATOR
2000	006510	012760	000011	000000		MOV	#11, RMCS1(R0)		CLEAR THE DRIVE
2001	006516	012760	000013	000000		MOV	#13, RMCS1(R0)		RELEASE THE DRIVE
2002	006524	104035				ERROR	35		TYPE ERROR MESSAGE 35
2003	006526	013737	001170	001126	68\$:	MOV	STMP2, \$BDDAT		LOOK FOR BIT FAILURES WHEN RMDS1 READ
2004	006534	013737	001224	001234		MOV	PORTA, PTNBR		CHANGE PORT NUMBER
2005	006542	042737	100000	001170		BIC	\$ATA, STMP2		DON'T CHECK THE ATTN BIT
2006	006550	023737	001124	001170		CMP	\$CDDAT, STMP2		ALL BITS OK ?
2007	006556	001401				BEQ	69\$		BR IF OK FROM PORT A.
2008	006560	104037				ERROR	37		REPORT ERROR
2009	006562	013737	001172	001126	69\$:	MOV	STMP3, \$BDDAT		CHECK RMDS1 FOR BIT FAILURES - FROM PORT B.
2010	006570	013737	001226	001234		MOV	PORTB, PTNBR		CHANGE PORT NUMBER
2011	006576	042737	100000	001172		BIC	\$ATA, STMP3		DON'T CHECK THE ATTN BIT
2012	006604	023737	001124	001172		CMP	\$CDDAT, STMP3		SEE IF READ OK FROM PORT B.
2013	006612	001401				BEQ	70\$		BR IF OK
2014	006614	104037				ERROR	37		REPORT THE ERROR
2015	006616	000240			70\$:	NOP			
2016	006620	000004			4\$:	SCOPE			;LOOP ?

2017									
2018									
2019									
2020									
2021									
2022									
2023									
2024									
2025									
2026									
2027									
2028									
2029									
2030									
2031									
2032									
2033									
2034	006622								
2035	006622	005737	001266			TST	KYBCTL		PERFORMING ONLY SINGLE TESTS ?
2036	006626	001406				BEQ	2\$		BR IF NOT
2037	006630	100002				BPL	1\$		BR IF JUST ENTERED TEST
2038	006632	000137	002424			JMP	EXEC		RETURN & GET NEXT TEST NUMBER
2039	006636	012737	177777	001266	1\$:	MOV	#-1, KYBCTL		SET SINGLE TEST INDICATOR
2040	006644	112737	000005	001102	2\$:	MOVB	#5, \$STNM		TEST NUMBER
2041	006652	012737	006674	001106		MOV	\$TEST5, \$LPADR		LOAD LOOP ON TEST ADDRESS

```

*****
*TEST 5      MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT B
*
*MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT B
*
*  A.  WRITE 0'S INTO RMDS1 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
*****

```


N03

CZRMHBD RM03/2 DU POR LGC 2
CZRMHBD.P11 21-NOV-77 13:34

MACY11 30(1046) 21-NOV-77 14:15 PAGE 39
TS MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT B

SEQ 0039

2042	006660	012737	006674	001110	MOV	#TEST5,\$LPERR	;LOAD LOOP ON ERROR ADDRESS
2043	006666	012737	000001	001176	MOV	#1,\$TIMES	;DO 1 ITERATION
2044	006674	012706	001100		TESTS: MOV	#STACK,\$P	;SETUP THE STACK POINTER
2045	006700	005037	001262		CLR	\$TIMEB	;CLEAR THE TIMEOUT VALUE STORAGE LOCATION
2046	006704	005037	001264		CLR	\$TIMEBP	;CLEAR THE + 25% TOLERANCE LOCATION
2047							
2048							
2049							
2050							
2051	006710	005037	001252		CLR	\$TIME	;CLEAR THE ELAPSED TIME COUNTER
2052	006714	012737	003720	001254	MOV	#2000,\$WATCH	;SET WATCH TO 2000 MS
2053							
2054							
2055							
2056							
2057							
2058	006722	113760	001226	000010	MOVB	\$PORTB,\$RMCS2(\$R0)	;SELECT PORT B
2059	006730	013737	001226	001236	MOV	\$PORTB,\$SEIZPT	;STORE SEIZING PORT'S ADDRESS
2060	006736	005060	000012		CLR	\$RMD\$1(\$R0)	;WRITE RMD\$1
2061	006742	113760	001224	000010	MOVB	\$PORTA,\$RMCS2(\$R0)	;SELECT PORT A
2062	006750	013737	001224	001234	MOV	\$PORTA,\$PTNBR	;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2063	006756	013737	001224	001240	MOV	\$PORTA,\$OPPR	;OPPOSITE PORT ADDRESS
2064	006764	016037	000012	001126	MOV	\$RMD\$1(\$R0),\$SBDDAT	;SEE IF DRIVE SEIZED BY PORT B
2065	006772	010037	001122		MOV	\$R0,\$SBADR	;R\$11 BASE ADDRESS
2066	006776	062737	000012	001122	ADD	\$RMD\$1,\$SBADR	;GENERATE BAD REGISTER ADDRESS
2067	007004	005037	001124		CLR	\$SGDDAT	;REGISTER SHOULD BE ZERO
2068	007010	023737	001124	001126	CMP	\$SGDDAT,\$SBDDAT	;IS THE REGISTER ZERO
2069	007016	001403			BEQ	64\$	;BR IF IT IS
2070	007020	104030			ERROR	30	;REPORT THE ERROR
2071	007022	000137	007536		JMP	4\$	;BYPASS REST OF THE SUBTEST
2072	007026						
2073	007026	113760	001226	000010	MOVB	\$PORTB,\$RMCS2(\$R0)	;SELECT PORT B
2074	007034	013737	001226	001234	MOV	\$PORTB,\$PTNBR	;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2075	007042	016037	000012	001126	MOV	\$RMD\$1(\$R0),\$SBDDAT	;SEE IF SEIZING PORT SEES CORRECT STATUS
2076	007050	012737	011700	001124	MOV	\$MOL!PGM!DPR!DRY!VV,\$SGDDAT	;EXPECTED STATUS
2077	007056	013737	001124	001166	MOV	\$SGDDAT,\$STMP1	;USE GOOD DATA AS A MASK
2078	007064	005137	001166		COM	\$STMP1	;COMPLEMENT THE EXPECTED STATUS
2079	007070	013737	001126	001164	MOV	\$SBDDAT,\$STMP0	;SAVE THE ACTUAL STATUS
2080	007076	043737	001166	001164	BIC	\$STMP1,\$STMP0	;CLEAR UNWANTED BITS
2081	007104	023737	001124	001164	CMP	\$SGDDAT,\$STMP0	;ARE THE EXPECTED STATUS BITS SET ?
2082	007112	001401			BEQ	65\$	;BR IF THEY ARE
2083	007114	104031			ERROR	31	;REPORT THE ERROR
2084	007116	000240			NOP		
2085							
2086							
2087							
2088							
2089	007120	113760	001224	000010	MOVB	\$PORTA,\$RMCS2(\$R0)	;SELECT PORT A
2090	007126	013737	001224	001234	MOV	\$PORTA,\$PTNBR	;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2091	007134	005760	000012		TST	\$RMD\$1(\$R0)	;WAIT FOR THE DRIVE TO TIMEOUT
2092	007140	001006			BNE	2\$	;BR WHEN TIMEOUT OCCURS
2093	007142	005737	001254		TST	\$WATCH	;CHECK WATCH
2094	007146	001372			BNE	1\$	;BR IF NOT ZERO
2095	007150	104006			ERROR	6	;NO TIMEOUT WITHIN 2 SECONDS
2096	007152	000137	007210		JMP	3\$	;BYPASS THE REST OF THE TEST
2097	007156	013737	001252	001262	MOV	\$TIME,\$TIMEB	;SAVE THE ELAPSED TIME FOR PORT B


```

2098 007164 004537 013432 JSR R5,TOLER ;CALCULATE THE TOLERANCE
2099 007170 001262 .WORD TIMEB ;TIMEOUT VALUE FOR PORT B
2100 007172 012637 001264 MOV (SP)+,TIMEBP ;+25% TOLERANCE
2101
2102 ;*****
2103 ;VERIFY THAT THE TIMEOUT ONE-SHOT VALUE IS AT LEAST 500 MS
2104
2105 007176 023727 001262 000764 CMP TIMEB,#500. ;IS TIMEOUT VALUE AT LEAST 500 MS ?
2106 007204 103001 BHS 3$ ;OR IF IT IS
2107 007206 104007 ERROR 7 ;TIMEOUT LESS THAN 500 MS
2108
2109 ;*****
2110 ;VERIFY THAT THE DRIVE RETURNED TO NEUTRAL AFTER PORT B TIMED OUT
2111
2112 007210 3$:
2113
2114 ;VERIFY THAT THE DRIVE IS IN NEUTRAL
2115
2116 007210 005037 001250 CLR RELERR ;CLEAR THE 'RELEASE ERROR' INDICATOR
2117 007214 012737 000012 001122 MOV #RMD51,$BDDADR ;FORM THE ADDRESS OF RMD51 FOR TYPEOUT
2118 007222 060037 001122 ADD R0,$BDDADR ;ADD THE I/O BASE ADDRESS
2119 007226 012737 011700 001124 MOV #MOL:PGM:DPR:DRY! VV,$GDDAT ;COMPARISON CONSTANT
2120 007234 113760 001224 000010 MOVB PORTA, RMCS2(R0) ;SELECT PORT A.
2121 007242 016037 000012 001170 MOV RMD51(R0), $TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
2122 007250 013737 001170 001164 MOV $TMP2, $TMP0 ;COPY IT INTO 'TMP0'
2123 007256 042737 100100 001164 BIC #ATA!VV, $TMP0 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2124 007264 113760 001226 000010 MOVB PORTB, RMCS2(R0) ;SELECT PORT B.
2125 007272 016037 000012 001172 MOV RMD51(R0), $TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
2126 007300 013737 001172 001166 MOV $TMP3, $TMP1 ;COPY IT INTO 'TMP1'
2127 007306 042737 100100 001166 BIC #ATA!VV, $TMP1 ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2128 007314 023737 001164 001166 CMP $TMP0, $TMP1 ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2129 007322 001006 BNE 66$ ;OR IF NOT
2130 007324 005737 001164 TST $TMP0 ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
2131 007330 001045 BNE 68$ ;OR IF NOT
2132 007332 104034 ERROR 34 ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2133 007334 000137 007534 JMP 70$ ;BYPASS THE REST OF THE CHECKS
2134 007340 013737 001170 001126 66$: MOV $TMP2, $BDDAT ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2135 007346 013737 001226 001234 MOV PORTB, PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2136 007354 113760 001226 000010 MOVB PORTB, RMCS2(R0) ;SELECT PORT B.
2137 007362 005737 001164 TST $TMP0 ;SEE IF STATUS EQ 0 FROM PORT A.
2138 007366 001414 BEQ 67$ ;OR IF ZERO
2139 007370 013737 001224 001234 MOV PORTA, PTNBR ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2140 007376 013737 001172 001126 MOV $TMP3, $BDDAT ;'BAD DATA' FOR ERROR TYPE OUT
2141 007404 113760 001224 000010 MOVB PORTA, RMCS2(R0) ;SELECT PORT A.
2142 007412 005737 001166 TST $TMP1 ;SEE IF STATUS EQ ZERO FROM PORT B.
2143 007416 001012 BNE 68$ ;OR IF NOT
2144 007420 012737 177777 001250 67$: MOV #-1, RELERR ;SET 'RELEASE ERROR' INDICATOR
2145 007426 012760 000011 000000 MOV #11, RMCS1(R0) ;CLEAR THE DRIVE
2146 007434 012760 000013 000000 MOV #13, RMCS1(R0) ;RELEASE THE DRIVE
2147 007442 104035 ERROR 35 ;TYPE ERROR MESSAGE 35
2148 007444 013737 001170 001126 68$: MOV $TMP2, $BDDAT ;LOOK FOR BIT FAILURES WHEN RMD51 READ
2149 007452 013737 001224 001234 MOV PORTA, PTNBR ;CHANGE PORT NUMBER
2150 007460 042737 100000 001170 BIC #ATA, $TMP2 ;DON'T CHECK THE ATTN BIT
2151 007466 023737 001124 001170 CMP $GDDAT, $TMP2 ;ALL BITS OK ?
2152 007474 001401 BEQ 69$ ;OR IF OK FROM PORT A.
2153 007476 104037 ERROR 37 ;REPORT ERROR

```


C04

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046)
TS21-NOV-77 14:15 PAGE 41
MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT B

SEQ 0041

```

2154 007500 013737 001172 001126 69$: MOV STMP3,SBDDAT ;CHECK RMD51 FOR BIT FAILURES - FROM PORT B.
2155 007506 013737 001226 001234 MOV PORTB,PTNBR ;CHANGE PORT NUMBER
2156 007514 042737 100000 001172 BIC #ATA,STMP3 ;DON'T CHECK THE ATTN BIT
2157 007522 023737 001124 001172 CMP SGDDAT,STMP3 ;SEE IF READ OK FROM PORT B.
2158 007530 001401 BEQ 70$ ;BR IF OK
2159 007532 104037 ERROR 37 ;REPORT THE ERROR
2160 007534 000240 70$: NOP
2161 007536 000004 4$: SCOPE ;LOOP ?

```

```

*****
*TEST 6 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 RMD51, 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 RMD51, AS READ THROUGH BOTH
* PORTS, ARE CORRECT.
*
* C. RETURN THE 'CONTROLLER SELECT' SWITCH TO THE 'A/B' POSITION. VERIFY
* THE DRIVE STATE.
*****

```

```

2182 007540
2183 007540 005737 001266
2184 007544 001406
2185 007546 100002
2186 007550 000137 002424
2187 007554 012737 177777 001266
2188 007562 112737 000006 001102
2189 007570 012737 007612 001106
2190 007576 012737 007612 001110
2191 007604 012737 000001 001176
2192 007612 012706 001100
2193
2194
2195
2196 007616 113760 001224 000010
2197 007624 005060 000012
2198 007630 012760 000011 000000
2199 007636 012760 000013 000000
2200 007644 113760 001226 000010
2201 007652 005060 000012
2202 007656 012760 000011 000000
2203 007664 012760 000013 000000
2204 007672 104401 017414
2205 007676 104401 017562
2206 007702 000000
2207
2208
2209

```

```

TST6: 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 #6,$STNM ;TEST NUMBER
MOV #TEST6,$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,RMCS2(R0) ;SELECT PORT #A
CLR RMD51(R0) ;SEIZE THE DRIVE
MOV #11,RMCS1(R0) ;ISSUE DRIVE CLEAR
MOV #13,RMCS1(R0) ;RELEASE THE DRIVE
MOVB PORTB,RMCS2(R0) ;SELECT PORT #B
CLR RMD51(R0) ;SEIZE THE DRIVE THROUGH PORT 'B'
MOV #11,RMCS1(R0) ;ISSUE DRIVE CLEAR
MOV #13,RMCS1(R0) ;RELEASE THE DRIVE
TYPE ,SWCHA ;SWITCH TO 'A'
TYPE ,CONTUE ;PRESS 'CONTINUE'
HALT

;VERIFY THAT THE DRIVE IS IN NEUTRAL

```

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34

MACY11 30(1046)
T6

21-NOV-77 14:15 PAGE 42

SEQ 0042

TEST 'CONTROLLER SELECT' SWITCH, DRIVE CYCLED UP

2210	007704	005037	001250		CLR	RELERR	; CLEAR THE 'RELEASE ERROR' INDICATOR
2211	007710	012737	000012	001122	MOV	#RMD51, \$BDADR	; FORM THE ADDRESS OF RMD51 FOR TYPEOUT
2212	007716	060037	001122		ADD	RO, \$BDADR	; ADD THE I/O BASE ADDRESS
2213	007722	012737	011700	001124	MOV	#MOL!PGM!DPR!DRY!VV, \$GDDAT	; COMPARISON CONSTANT
2214	007730	113760	001224	000010	MOVB	PORTA, RMCS2(RO)	; SELECT PORT A.
2215	007736	016037	000012	001170	MOV	RMD51(RO), \$TMP2	; GET THE DRIVE STATUS REGISTER FROM PORT A.
2216	007744	013737	001170	001164	MOV	\$TMP2, \$TMP0	; COPY IT INTO 'TMP0'
2217	007752	042737	100100	001164	BIC	#ATA!VV, \$TMP0	; CLEAR PORT DEPENDENT BITS FROM THE COPY
2218	007760	113760	001226	000010	MOVB	PORTB, RMCS2(RO)	; SELECT PORT B.
2219	007766	016037	000012	001172	MOV	RMD51(RO), \$TMP3	; GET THE DRIVE STATUS REGISTER FROM PORT B.
2220	007774	013737	001172	001166	MOV	\$TMP3, \$TMP1	; COPY IT INTO 'TMP1'
2221	010002	042737	100100	001166	BIC	#ATA!VV, \$TMP1	; CLEAR PORT DEPENDENT BITS FROM THE COPY
2222	010010	023737	001164	001166	CMP	\$TMP0, \$TMP1	; IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2223	010016	001006			BNE	64\$	; BR IF NOT
2224	010020	005737	001164		TST	\$TMP0	; REGISTERS ARE THE SAME: ARE THEY ZERO ?
2225	010024	001045			BNE	66\$	; BR IF NOT
2226	010026	104034			ERROR	34	; REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2227	010030	000137	010230		JMP	68\$	; BYPASS THE REST OF THE CHECKS
2228	010034	013737	001170	001126	MOV	\$TMP2, \$BDAT	; SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2229	010042	013737	001226	001234	MOV	PORTB, PTNBR	; SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2230	010050	113760	001226	000010	MOVB	PORTB, RMCS2(RO)	; SELECT PORT B.
2231	010056	005737	001164		TST	\$TMP0	; SEE IF STATUS EQ 0 FROM PORT A.
2232	010062	001414			BEQ	65\$	; BR IF ZERO
2233	010064	013737	001224	001234	MOV	PORTA, PTNBR	; SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2234	010072	013737	001172	001126	MOV	\$TMP3, \$BDAT	; 'BAD DATA' FOR ERROR TYPE OUT
2235	010100	113760	001224	000010	MOVB	PORTA, RMCS2(RO)	; SELECT PORT A.
2236	010106	005737	001166		TST	\$TMP1	; SEE IF STATUS EQ ZERO FROM PORT B.
2237	010112	001012			BNE	66\$	; BR IF NOT
2238	010114	012737	177777	001250	MOV	#-1, RELERR	; SET 'RELEASE ERROR' INDICATOR
2239	010122	012760	000011	000000	MOV	#11, RMCS1(RO)	; CLEAR THE DRIVE
2240	010130	012760	000013	000000	MOV	#13, RMCS1(RO)	; RELEASE THE DRIVE
2241	010136	104017			ERROR	17	; TYPE ERROR MESSAGE 17
2242	010140	013737	001170	001126	MOV	\$TMP2, \$BDAT	; LOOK FOR BIT FAILURES WHEN RMD51 READ
2243	010146	013737	001224	001234	MOV	PORTA, PTNBR	; CHANGE PORT NUMBER
2244	010154	042737	100000	001170	BIC	#ATA, \$TMP2	; DON'T CHECK THE ATTN BIT
2245	010162	023737	001124	001170	CMP	\$GDDAT, \$TMP2	; ALL BITS OK ?
2246	010170	001401			BEQ	67\$	; BR IF OK FROM PORT A.
2247	010172	104037			ERROR	37	; REPORT ERROR
2248	010174	013737	001172	001126	MOV	\$TMP3, \$BDAT	; CHECK RMD51 FOR BIT FAILURES - FROM PORT B.
2249	010202	013737	001226	001234	MOV	PORTB, PTNBR	; CHANGE PORT NUMBER
2250	010210	042737	100000	001172	BIC	#ATA, \$TMP3	; DON'T CHECK THE ATTN BIT
2251	010216	023737	001124	001172	CMP	\$GDDAT, \$TMP3	; SEE IF READ OK FROM PORT B.
2252	010224	001401			BEQ	68\$	; BR IF OK
2253	010226	104037			ERROR	37	; REPORT THE ERROR
2254	010230	000240			NOP		
2255	010232	104401	017477		TYPE	, SWTCHB	; SWITCH TO 'B'
2256	010236	104401	017562		TYPE	, CONTUE	; PRESS 'CONTINUE'
2257	010242	000000			HALT		
2258							
2259							
2260							; VERIFY THAT THE DRIVE IS IN NEUTRAL
2261	010244	005037	001250		CLR	RELERR	; CLEAR THE 'RELEASE ERROR' INDICATOR
2262	010250	012737	000012	001122	MOV	#RMD51, \$BDADR	; FORM THE ADDRESS OF RMD51 FOR TYPEOUT
2263	010256	060037	001122		ADD	RO, \$BDADR	; ADD THE I/O BASE ADDRESS
2264	010262	012737	011700	001124	MOV	#MOL!PGM!DPR!DRY!VV, \$GDDAT	; COMPARISON CONSTANT
2265	010270	113760	001224	000010	MOVB	PORTA, RMCS2(RO)	; SELECT PORT A.

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046)
T621-NOV-77 14:15 PAGE 43
TEST 'CONTROLLER SELECT'

SWITCH, DRIVE CYCLED UP

SEQ 0043

2266	010276	016037	000012	001170		MOV	RMDS1(R0),STMP2	GET THE DRIVE STATUS REGISTER FROM PORT A.
2267	010304	013737	001170	001164		MOV	STMP2,STMP0	COPY IT INTO 'STMP0'
2268	010312	042737	100100	001164		BIC	#ATA!VV,STMP0	CLEAR PORT DEPENDENT BITS FROM THE COPY
2269	010320	113760	001226	000010		MOVB	PORTB,RMCS2(R0)	SELECT PORT B.
2270	010326	016037	000012	001172		MOV	RMDS1(R0),STMP3	GET THE DRIVE STATUS REGISTER FROM PORT B.
2271	010334	013737	001172	001166		MOV	STMP3,STMP1	COPY IT INTO 'STMP1'
2272	010342	042737	100100	001166		BIC	#ATA!VV,STMP1	CLEAR PORT DEPENDENT BITS FROM THE COPY
2273	010350	023737	001164	001166		CMP	STMP0,STMP1	IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2274	010356	001006				BNE	69\$	BR IF NOT
2275	010360	005737	001164			TST	STMP0	REGISTERS ARE THE SAME: ARE THEY ZERO ?
2276	010364	001045				BNE	71\$	BR IF NOT
2277	010366	104034				ERROR	34	REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2278	010370	000137	010570			JMP	73\$	BYPASS THE REST OF THE CHECKS
2279	010374	013737	001170	001126	69\$:	MOV	STMP2,\$BDDAT	SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2280	010402	013737	001226	001234		MOV	PORTB,PTNBR	SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2281	010410	113760	001226	000010		MOVB	PORTB,RMCS2(R0)	SELECT PORT B.
2282	010416	005737	001164			TST	STMP0	SEE IF STATUS EQ 0 FROM PORT A.
2283	010422	001414				BEQ	70\$	BR IF ZERO
2284	010424	013737	001224	001234		MOV	PORTA,PTNBR	SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2285	010432	013737	001172	001126		MOV	STMP3,\$BDDAT	'BAD DATA' FOR ERROR TYPE OUT
2286	010440	113760	001224	000010		MOVB	PORTA,RMCS2(R0)	SELECT PORT A.
2287	010446	005737	001166			TST	STMP1	SEE IF STATUS EQ ZERO FROM PORT B.
2288	010452	001012				BNE	71\$	BR IF NOT
2289	010454	012737	177777	001250	70\$:	MOV	#-1,RELEA	SET 'RELEASE ERROR' INDICATOR
2290	010462	012760	000011	000000		MOV	#11,RMCS1(R0)	CLEAR THE DRIVE
2291	010470	012760	000013	000000		MOV	#13,RMCS1(R0)	RELEASE THE DRIVE
2292	010476	104020				ERROR	20	TYPE ERROR MESSAGE 20
2293	010500	013737	001170	001126	71\$:	MOV	STMP2,\$BDDAT	LOOK FOR BIT FAILURES WHEN RMDS1 READ
2294	010506	013737	001224	001234		MOV	PORTA,PTNBR	CHANGE PORT NUMBER
2295	010514	042737	100000	001170		BIC	#ATA,STMP2	DON'T CHECK THE ATTN BIT
2296	010522	023737	001124	001170		CMP	\$GDDAT,STMP2	ALL BITS OK ?
2297	010530	001401				BEQ	72\$	BR IF OK FROM PORT A.
2298	010532	104037				ERROR	37	REPORT ERROR
2299	010534	013737	001172	001126	72\$:	MOV	STMP3,\$BDDAT	CHECK RMDS1 FOR BIT FAILURES - FROM PORT B.
2300	010542	013737	001226	001234		MOV	PORTB,PTNBR	CHANGE PORT NUMBER
2301	010550	042737	100000	001172		BIC	#ATA,STMP3	DON'T CHECK THE ATTN BIT
2302	010556	023737	001124	001172		CMP	\$GDDAT,STMP3	SEE IF READ OK FROM PORT B.
2303	010564	001401				BEQ	73\$	BR IF OK
2304	010566	104037				ERROR	37	REPORT THE ERROR
2305	010570	000240			73\$:	NOP		
2306	010572	005737	001266			TST	KYBCTL	SINGLE TEST MODE ?
2307	010576	001402				BEQ	1\$	BR IF NOT
2308	010600	104401	017325			TYPE	,SWTCHN	RETURN SWITCH TO 'A/B'
2309	010604	000004			1\$:	SCOPE		LOOP ?

```

*****
*TEST 7      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 RMDS1, AS READ THROUGH BOTH

```

F04

CZRMHBO RM03/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 44
T7 TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT A

SEQ 0044

2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377010606 005737 001266
010606 001406
010612 100002
010614 000137 002424
010616 012737 177777 001266
010622 112737 000007 001102
010630 012737 010660 001106
010636 012737 010660 001110
010644 012737 000001 001176
010652 012706 001100
010660 113760 001224 000010
010672 013737 001224 001234
010700 104401 017631

010704 032760 010000 000012
010712 001374
010714 104401 017414
010720 104401 017653

010724 032760 010000 000012
010732 001774

010734 005037 001244
010740 016037 000012 001126
010746 012737 000012 001122
010754 060037 001122
010760 012737 110600 001124

```

; * 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' SETS WHEN ATTEMPTING TO ACCESS THE DRIVE THROUGH
; * PORT B. ATTEMPT TO SET PORT REQUEST BY WRITING 0'S
; * INTO RMD51 THROUGH PORT B.
; * G. ISSUE A RELEASE COMMAND THROUGH PORT A. VERIFY THAT THE
; * DRIVE REMAINS LOCKED ON PORT A.
; *
; *****
TST7: 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
      MOV #1, KYBCTL ;SET SINGLE TEST INDICATOR
      MOV #7, $STNM ;TEST NUMBER
      MOV #TEST7, $LPADR ;LOAD LOOP ON TEST ADDRESS
      MOV #TEST7, $LPERR ;LOAD LOOP ON ERROR ADDRESS
      MOV #1, $TIMES ;DO 1 ITERATION
      MOV #STACK, SP ;SETUP THE STACK POINTER
      MOV PORTA, RMCS2(R0) ;SELECT PORT A
      MOV PORTA, PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
      TYPE ,CYCLED ;'CYCLE DOWN THE DRIVE'
; *****
;WAIT FOR 'MOL' TO RESET
15: BIT #MOL, RMD51(R0) ;TEST 'MOL'
     BNE 15 ;BR IF IT IS STILL SET
     TYPE ,SWTCH ;SWITCH TO 'A'
     TYPE ,CYCLEU ;'CYCLE UP THE DRIVE'
; *****
;WAIT FOR DRIVE TO CYCLE UP AFTER SWITCH CHANGED
25: BIT #MOL, RMD51(R0) ;TEST 'MOL' AGAIN
     BEQ 25 ;BR IF NOT SET
; *****
;DRIVE IS CYCLED UP, CHECK STATUS THROUGH PORT A
      CLR CKERR ;CLEAR THE 'CHECK ERROR' INDICATOR
      MOV RMD51(R0), $BDDAT ;GET CONTENTS OF RMD51
      MOV #RMD51, $BDADR ;FORM REGISTER ADDRESS OF ERROR MESSAGE
      ADD R0, $BDADR ;ADD R011 BASE ADDRESS
      MOV #ATA!MOL!DPR!DRY, $GDDAT ;WHAT REGISTER SHOULD BE

```


G04

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046)
T7

21-NOV-77 14:15 PAGE 45

SEQ 0045

TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT A

2378	010766	013737	001126	001164	MOV	\$BDDAT,STMP0	;MOVE REGISTER CONTENTS TO 'STMP0'
2379	010774	042737	066077	001164	BIC	#1C111700,STMP0	;SAVE SPECIFIED BITS
2380	011002	023737	001124	001164	CMP	\$GDDAT,STMP0	;COMPARE THE BITS
2381	011010	001414			BEQ	64\$	;BR IF OK
2382	011012	013737	001126	001174	MOV	\$BDDAT,STMP4	;COPY 'BAD DATA'
2383	011020	042737	111700	001174	BIC	#111700,STMP4	;CLEAR THE MASKED BITS
2384	011026	053737	001174	001124	BIS	STMP4,\$GDDAT	; 'OR' WITH GOOD DATA FOR TYPEOUT
2385	011034	104021			ERROR	21	;TYPE MESSAGE 21
2386	011036	005137	001244		COM	CKERR	;SET THE REGISTER COMPARE ERROR INDICATOR
2387	011042	000240			64\$: NOP		
2388							
2389							
2390							
2391							
2392	011044	012760	000011	000000	MOV	#11,RMCS1(RO)	;ISSUE A DRIVE CLEAR
2393	011052	012760	000021	000000	MOV	#21,RMCS1(RO)	;ISSUE A READIN PRESET
2394	011060	012760	010000	000032	MOV	#FMT16,RMOF(RO)	;SET FMT16
2395							
2396							
2397							
2398							
2399							
2400	011066	113760	001226	000010	MOV	PORTB,RMCS2(RO)	;SELECT PORT B
2401	011074	013737	001226	001234	MOV	PORTB,PTNBR	;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2402	011102	005037	001244		CLR	CKERR	;CLEAR THE 'CHECK ERROR' INDICATOR
2403	011106	016037	000012	001126	MOV	RMDS1(RO), \$BDDAT	;GET CONTENTS OF RMDS1
2404	011114	012737	000012	001122	MOV	#RMDS1,\$BDDADR	;FORM REGISTER ADDRESS OF ERROR MESSAGE
2405	011122	060037	001122		ADD	RO,\$BDDADR	;ADD RM11 BASE ADDRESS
2406	011126	005037	001124		CLR	\$GDDAT	;WHAT REGISTER SHOULD BE
2407	011132	013737	001126	001164	MOV	\$BDDAT,STMP0	;MOVE REGISTER CONTENTS TO 'STMP0'
2408	011140	042737	000077	001164	BIC	#1C177700,STMP0	;SAVE SPECIFIED BITS
2409	011146	023737	001124	001164	CMP	\$GDDAT,STMP0	;COMPARE THE BITS
2410	011154	001414			BEQ	66\$	;BR IF OK
2411	011156	013737	001126	001174	MOV	\$BDDAT,STMP4	;COPY 'BAD DATA'
2412	011164	042737	177700	001174	BIC	#177700,STMP4	;CLEAR THE MASKED BITS
2413	011172	053737	001174	001124	BIS	STMP4,\$GDDAT	; 'OR' WITH GOOD DATA FOR TYPEOUT
2414	011200	104022			ERROR	22	;TYPE MESSAGE 22
2415	011202	005137	001244		COM	CKERR	;SET THE REGISTER COMPARE ERROR INDICATOR
2416	011206	000240			66\$: NOP		
2417	011210	005037	001244		CLR	CKERR	;CLEAR THE 'CHECK ERROR' INDICATOR
2418	011214	016037	000010	001126	MOV	RMCS2(RO), \$BDDAT	;GET CONTENTS OF RMCS2
2419	011222	012737	000010	001122	MOV	#RMCS2,\$BDDADR	;FORM REGISTER ADDRESS OF ERROR MESSAGE
2420	011230	060037	001122		ADD	RO,\$BDDADR	;ADD RM11 BASE ADDRESS
2421	011234	012737	010000	001124	MOV	#NED,\$GDDAT	;WHAT REGISTER SHOULD BE
2422	011242	013737	001126	001164	MOV	\$BDDAT,STMP0	;MOVE REGISTER CONTENTS TO 'STMP0'
2423	011250	042737	167777	001164	BIC	#1CNEED,STMP0	;SAVE SPECIFIED BITS
2424	011256	023737	001124	001164	CMP	\$GDDAT,STMP0	;COMPARE THE BITS
2425	011264	001414			BEQ	68\$	;BR IF OK
2426	011266	013737	001126	001174	MOV	\$BDDAT,STMP4	;COPY 'BAD DATA'
2427	011274	042737	010000	001174	BIC	#NED,STMP4	;CLEAR THE MASKED BITS
2428	011302	053737	001174	001124	BIS	STMP4,\$GDDAT	; 'OR' WITH GOOD DATA FOR TYPEOUT
2429	011310	104023			ERROR	23	;TYPE MESSAGE 23
2430	011312	005137	001244		COM	CKERR	;SET THE REGISTER COMPARE ERROR INDICATOR
2431	011316	000240			68\$: NOP		
2432	011320	005060	000012		CLR	RMDS1(RO)	;TRY TO SET REQUEST BY WRITING THROUGH
2433							;THE LOCKED OUT PORT (PORT 'B')

H04

CZRMHBO RM03/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 46
T7 TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT A

SEQ 0046

```

2434
2435
2436
2437
2438 011324 113760 001224 000010      MOVB   PORTA, RMCS2(R0) ; SELECT PORT A
2439 011332 013737 001224 001234      MOV    PORTA, PTNBR ; MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2440 011340 012760 000013 000012      MOV    #13, RMDS1(R0) ; ISSUE A RELEASE THROUGH PORT A
2441 011346 013737 001224 001236      MOV    PORTA, SEIZPT ; ADDRESS OF 'LOCKED ON' PORT
2442 011354 113760 001226 000010      MOVB   PORTB, RMCS2(R0) ; SELECT PORT B
2443 011362 013737 001226 001234      MOV    PORTB, PTNBR ; MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2444 011370 005037 001244      CLR    CKERR ; CLEAR THE 'CHECK ERROR' INDICATOR
2445 011374 016037 000012 001126      MOV    RMDS1(R0), $BDDAT ; GET CONTENTS OF RMDS1
2446 011402 012737 000012 001122      MOV    $RMDS1, $BDAOR ; FORM REGISTER ADDRESS OF ERROR MESSAGE
2447 011410 060037 001122      ADD    R0, $BDAOR ; ADD RH11 BASE ADDRESS
2448 011414 005037 001124      CLR    $GDDAT ; WHAT REGISTER SHOULD BE
2449 011420 013737 001126 001164      MOV    $BDDAT, $TMP0 ; MOVE REGISTER CONTENTS TO 'TMP0'
2450 011426 042737 000077 001164      BIC    #177700, $TMP0 ; SAVE SPECIFIED BITS
2451 011434 023737 001124 001164      CMP    $GDDAT, $TMP0 ; COMPARE THE BITS
2452 011442 001414      BEQ    70$ ; BR IF OK
2453 011444 013737 001126 001174      MOV    $BDDAT, $TMP4 ; COPY 'BAD DATA'
2454 011452 042737 177700 001174      BIC    #177700, $TMP4 ; CLEAR THE MASKED BITS
2455 011460 053737 001174 001124      BIS    $TMP4, $GDDAT ; 'OR' WITH GOOD DATA FOR TYPEOUT
2456 011466 104024      ERROR 24 ; TYPE MESSAGE 24
2457 011470 005137 001244      COM    CKERR ; SET THE REGISTER COMPARE ERROR INDICATOR
2458 011474 000240      NOP
2459
2460
2461
2462 011476 105737 001103      TSTB   $ERFLG ; DID AN ERROR OCCUR
2463 011502 001412      BEQ    3$ ; BR IF NOT
2464 011504 032777 001000 167426      BIT    #SW09, $SWR ; SEE IF LOOP ON ERROR (SWR9=1)
2465 011512 001406      BEQ    3$ ; BR IF NOT
2466 011514 105037 001103      CLRB   $ERFLG ; CLEAR THE ERROR FLAG
2467 011520 005037 001176      CLR    $TIMES ; CLEAR THE MAX ITERATION COUNT
2468 011524 000177 167360      JMP    $SLPERR ; GO TO THE LOOP ADDRESS
2469 011530 005737 001266      3$: TST    KYBCTL ; IN SINGLE TEST MODE ?
2470 011534 001460      BEQ    6$ ; BR IF NOT
2471 011536 032777 040000 167374      BIT    #SW14, $SWR ; LOOP ON TEST ?
2472 011544 001054      BNE    6$ ; BR IF LOOPING
2473 011546 104401 017631      TYPE   'CYCLED ; TYPE 'CYCLE DOWN'
2474 011552 104401 017325      TYPE   'SWTCHN ; 'SWITCH TO A/B'
2475 011556 104401 017653      TYPE   'CYCLEU ; 'CYCLE THE DRIVE UP'
2476 011562 113760 001224 000010      MOVB   PORTA, RMCS2(R0) ; SELECT PORT A
2477 011570 013737 001224 001234      MOV    PORTA, PTNBR ; MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2478 011576 032760 010000 000012      4$: BIT    $MOL, RMDS1(R0) ; CHECK 'MOL'
2479 011604 001374      BNE    4$ ; BR IF SET (DRIVE NOT CYCLED DOWN)
2480 011606 032760 010000 000012      5$: BIT    $MOL, RMDS1(R0) ; CHECK 'MOL' AGAIN
2481 011614 001774      BEQ    5$ ; BR IF NOT SET (DRIVE NOT CYCLED UP)
2482
2483
2484
2485
2486 011616 012760 000011 000000      MOV    #11, RMCS1(R0) ; ISSUE A DRIVE CLEAR THROUGH PORT A
2487 011624 012760 000021 000000      MOV    #21, RMCS1(R0) ; ISSUE A READIN PRESET THROUGH PORT A
2488 011632 012760 000013 000000      MOV    #13, RMCS1(R0) ; RELEASE PORT A
2489 011640 113760 001226 000010      MOVB   PORTB, RMCS2(R0) ; SELECT PORT B

```

```

;*****
;VERIFY THAT DRIVE STAYS LOCKED ON PORT A

```

```

; IF ERROR OCCURRED, CHECK FOR LOOP ON TEST

```

```

;*****
;SET VOLUME VALID FOR BOTH PORTS

```


CZRMHBO RMO3/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34

MACY11 30(1046)
T7

21-NOV-77 14:15 PAGE 47

SEQ 0047

TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT A

2490	011646	013737	001226	001234	MOV	PORTB,PTNBR	;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2491	011654	012760	000021	000000	MOV	#21,RMCS1(RO)	;ISSUE A READIN PRESET THROUGH PORT B
2492	011662	012760	010000	000032	MOV	#FMT16,RMOF(RO)	;SET FMT16
2493	011670	012760	000013	000000	MOV	#13,RMCS1(RO)	;RELEASE PORT B
2494	011676	104401	001207		6S: TYPE	SCALF	;CR-LF
2495	011702	012737	011610	001254	MOV	#5000.,WATCH	;SPINDLE MOTOR 'COOL DOWN' DELAY
2496	011710	005737	001254		7S: TST	WATCH	;FINISHED ?
2497	011714	001375			BNE	7S	;BR IF NOT
2498	011716	000004			SCOPE		;LOOP ?
2499	011720	000400			BR	TST10	;GO TO NEXT TEST

```

*****
*TEST 10      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 RMD51, 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' SETS WHEN ATTEMPTING TO ACCESS THE DRIVE THROUGH
*      PORT A.  ATTEMPT TO SET PORT REQUEST BY WRITING 0'S
*      INTO RMD51 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.
*****

```

2535	011722				TST10:	TST	KYBCTL	;PERFORMING ONLY SINGLE TESTS ?
2536	011722	005737	001266			BEG	2S	;BR IF NOT
2537	011726	001406				BPL	1S	;BR IF JUST ENTERED TEST
2538	011730	100002				JMP	EXEC	;RETURN & GET NEXT TEST NUMBER
2539	011732	000137	002424		1S:	MOV	#-1,KYBCTL	;SET SINGLE TEST INDICATOR
2540	011736	012737	177777	001266	2S:	MOVB	#10,STSTN	;TEST NUMBER
2541	011744	112737	000010	001102		MOV	#TEST10,SLPADR	;LOAD LOOP ON TEST ADDRESS
2542	011752	012737	011774	001106		MOV	#TEST10,SLPERR	;LOAD LOOP ON ERROR ADDRESS
2543	011760	012737	011774	001110		MOV	#1,STIMES	;DO 1 ITERATION
2544	011766	012737	000001	001176	TEST10:	MOV	#STACK,SP	;SETUP THE STACK POINTER
2545	011774	012706	001100					

```

2546 012000 113760 001226 000010      MOVB    PORTB, RMCS2(R0) ; SELECT PORT B
2547 012006 013737 001226 001234      MOV     PORTB, PTNBR ; MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2548 012014 104401 017631                TYPE    , CYCLED ; 'CYCLE DOWN THE DRIVE'
2549
2550      ;*****
2551      ;WAIT FOR 'MOL' TO RESET
2552
2553 012020 032760 010000 000012 1$:      BIT     #MOL, RMDS1(R0) ; TEST 'MOL'
2554 012026 001374                BNE     1$ ; BR IF IT IS STILL SET
2555 012030 104401 017477                TYPE    , SWCHB ; SWITCH TO 'B'
2556 012034 104401 017653                TYPE    , CYCLEU ; 'CYCLE UP THE DRIVE'
2557
2558      ;*****
2559      ;WAIT FOR DRIVE TO CYCLE UP AFTER SWITCH CHANGED
2560
2561 012040 032760 010000 000012 2$:      BIT     #MOL, RMDS1(R0) ; TEST 'MOL' AGAIN
2562 012046 001774                BEQ     2$ ; BR IF NOT SET
2563
2564      ;*****
2565      ;DRIVE IS CYCLED UP, CHECK STATUS THROUGH PORT B
2566
2567 012050 005037 001244                CLR     CKERR ; CLEAR THE 'CHECK ERROR' INDICATOR
2568 012054 016037 000012 001126      MOV     RMDS1(R0), $BDDAT ; GET CONTENTS OF RMDS1
2569 012062 012737 000012 001122      MOV     #RMDS1, $BDAADR ; FORM REGISTER ADDRESS OF ERROR MESSAGE
2570 012070 060037 001122                ADD     R0, $BDAADR ; ADD RH11 BASE ADDRESS
2571 012074 012737 110600 001124      MOV     #ATA!MOL!DPR!DRY, $GDDAT ; WHAT REGISTER SHOULD BE
2572 012102 013737 001126 001164      MOV     $BDDAT, $TMP0 ; MOVE REGISTER CONTENTS TO '$TMP0'
2573 012110 042737 066077 001164      BIC     #1C111700, $TMP0 ; SAVE SPECIFIED BITS
2574 012116 023737 001124 001164      CMP     $GDDAT, $TMP0 ; COMPARE THE BITS
2575 012124 001414                BEQ     64$ ; BR IF OK
2576 012126 013737 001126 001174      MOV     $BDDAT, $TMP4 ; COPY 'BAD DATA'
2577 012134 042737 111700 001174      BIC     #111700, $TMP4 ; CLEAR THE MASKED BITS
2578 012142 053737 001174 001124      BIS     $TMP4, $GDDAT ; 'OR' WITH GOOD DATA FOR TYPEOUT
2579 012150 104021                ERROR    21 ; TYPE MESSAGE 21
2580 012152 005137 001244                COM     CKERR ; SET THE REGISTER COMPARE ERROR INDICATOR
2581 012156 000240 64$:                NOP
2582
2583      ;*****
2584      ;SET VOLUME VALID FOR PORT B
2585
2586 012160 012760 000011 000000      MOV     #11, RMCS1(R0) ; ISSUE A DRIVE CLEAR
2587 012166 012760 000021 000000      MOV     #21, RMCS1(R0) ; ISSUE A READIN PRESET
2588 012174 012760 010000 000032      MOV     #FMT16, RMOF(R0) ; SET FMT16
2589
2590      ;*****
2591      ;CHECK THE DRIVE STATUS THROUGH PORT A; VERIFY THAT 'NED'
2592      ;SETS WHEN THE DRIVE IS ACCESSED THROUGH PORT A.
2593
2594 012202 113760 001224 000010      MOVB    PORTA, RMCS2(R0) ; SELECT PORT A
2595 012210 013737 001224 001234      MOV     PORTA, PTNBR ; MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2596 012216 005037 001244                CLR     CKERR ; CLEAR THE 'CHECK ERROR' INDICATOR
2597 012222 016037 000012 001126      MOV     RMDS1(R0), $BDDAT ; GET CONTENTS OF RMDS1
2598 012230 012737 000012 001122      MOV     #RMDS1, $BDAADR ; FORM REGISTER ADDRESS OF ERROR MESSAGE
2599 012236 060037 001122                ADD     R0, $BDAADR ; ADD RH11 BASE ADDRESS
2600 012242 005037 001124                CLR     $GDDAT ; WHAT REGISTER SHOULD BE
2601 012246 013737 001126 001164      MOV     $BDDAT, $TMP0 ; MOVE REGISTER CONTENTS TO '$TMP0'

```


K04

CZRMHBO RM03/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34MACY11 30(1046)
T1021-NOV-77 14:15 PAGE 49
TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT B

SEQ 0049

2602	012254	042737	000077	001164	BIC	#1C177700,STMP0	:SAVE SPECIFIED BITS
2603	012262	023737	001124	001164	CMP	\$GDDAT,STMP0	:COMPARE THE BITS
2604	012270	001414			BEQ	66\$	:BR IF OK
2605	012272	013737	001126	001174	MOV	\$BDDAT,STMP4	:COPY 'BAD DATA'
2606	012300	042737	177700	001174	BIC	#177700,STMP4	:CLEAR THE MASKED BITS
2607	012306	053737	001174	001124	BIS	STMP4,\$GDDAT	: 'OR' WITH GOOD DATA FOR TYPEOUT
2608	012314	104022			ERROR	22	:TYPE MESSAGE 22
2609	012316	005137	001244		COM	CKERR	:SET THE REGISTER COMPARE ERROR INDICATOR
2610	012322	000240			66\$: NOP		
2611	012324	005037	001244		CLR	CKERR	:CLEAR THE 'CHECK ERROR' INDICATOR
2612	012330	016037	000010	001126	MOV	RMCS2(R0), \$BDDAT	:GET CONTENTS OF RMCS2
2613	012336	012737	000010	001122	MOV	\$RMCS2, \$BDDAT	:FORM REGISTER ADDRESS OF ERROR MESSAGE
2614	012344	060037	001122		ADD	R0, \$BDDAT	:ADD RHI1 BASE ADDRESS
2615	012350	012737	010000	001124	MOV	\$NED, \$GDDAT	:WHAT REGISTER SHOULD BE
2616	012356	013737	001126	001164	MOV	\$BDDAT, STMP0	:MOVE REGISTER CONTENTS TO 'STMP0'
2617	012364	042737	167777	001164	BIC	#1C177700,STMP0	:SAVE SPECIFIED BITS
2618	012372	023737	001124	001164	CMP	\$GDDAT,STMP0	:COMPARE THE BITS
2619	012400	001414			BEQ	68\$	:BR IF OK
2620	012402	013737	001126	001174	MOV	\$BDDAT,STMP4	:COPY 'BAD DATA'
2621	012410	042737	010000	001174	BIC	\$NED,STMP4	:CLEAR THE MASKED BITS
2622	012416	053737	001174	001124	BIS	STMP4,\$GDDAT	: 'OR' WITH GOOD DATA FOR TYPEOUT
2623	012424	104023			ERROR	23	:TYPE MESSAGE 23
2624	012426	005137	001244		COM	CKERR	:SET THE REGISTER COMPARE ERROR INDICATOR
2625	012432	000240			68\$: NOP		
2626	012434	005060	000012		CLR	RMDS1(R0)	:TRY TO SET REQUEST BY WRITING THROUGH
2627							:THE LOCKED OUT PORT (PORT 'A')
2628							
2629							
2630							
2631							
2632	012440	113760	001226	000010	MOV	PORTB, RMCS2(R0)	:SELECT PORT B
2633	012446	013737	001226	001234	MOV	PORTB, PTNBR	:MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2634	012454	012760	000013	000012	MOV	#13, RMDS1(R0)	:ISSUE A RELEASE THROUGH PORT B
2635	012462	013737	001226	001236	MOV	PORTB, SEIZPT	:ADDRESS OF 'LOCKED ON' PORT
2636	012470	113760	001224	000010	MOV	PORTA, RMCS2(R0)	:SELECT PORT A
2637	012476	013737	001224	001234	MOV	PORTA, PTNBR	:MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2638	012504	005037	001244		CLR	CKERR	:CLEAR THE 'CHECK ERROR' INDICATOR
2639	012510	016037	000012	001126	MOV	RMDS1(R0), \$BDDAT	:GET CONTENTS OF RMDS1
2640	012516	012737	000012	001122	MOV	\$RMDS1, \$BDDAT	:FORM REGISTER ADDRESS OF ERROR MESSAGE
2641	012524	060037	001122		ADD	R0, \$BDDAT	:ADD RHI1 BASE ADDRESS
2642	012530	005037	001124		CLR	\$GDDAT	:WHAT REGISTER SHOULD BE
2643	012534	013737	001126	001164	MOV	\$BDDAT, STMP0	:MOVE REGISTER CONTENTS TO 'STMP0'
2644	012542	042737	000077	001164	BIC	#1C177700,STMP0	:SAVE SPECIFIED BITS
2645	012550	023737	001124	001164	CMP	\$GDDAT,STMP0	:COMPARE THE BITS
2646	012556	001414			BEQ	70\$	:BR IF OK
2647	012560	013737	001126	001174	MOV	\$BDDAT,STMP4	:COPY 'BAD DATA'
2648	012566	042737	177700	001174	BIC	#177700,STMP4	:CLEAR THE MASKED BITS
2649	012574	053737	001174	001124	BIS	STMP4,\$GDDAT	: 'OR' WITH GOOD DATA FOR TYPEOUT
2650	012602	104024			ERROR	24	:TYPE MESSAGE 24
2651	012604	005137	001244		COM	CKERR	:SET THE REGISTER COMPARE ERROR INDICATOR
2652	012610	000240			70\$: NOP		
2653							
2654							
2655							
2656	012612	105737	001103		TSTB	\$ERFLG	:DID AN ERROR OCCUR
2657	012616	001412			BEQ	3\$	:BR IF NOT

:VERIFY THAT DRIVE STAYS LOCKED ON PORT B

:IF ERROR OCCURRED, CHECK FOR LOOP ON TEST

L04

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046)
T10

21-NOV-77 14:15 PAGE 50

SEQ 0050

TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT B

```

2658 012620 032777 001000 166312 BIT #SW09,2SWR ;SEE IF LOOP ON ERROR (SWR9=1)
2659 012626 001406 BEQ 3$ ;BR IF NOT
2660 012630 105037 001103 CLRB $ERFLG ;CLEAR THE ERROR FLAG
2661 012634 005037 001176 CLR $TIMES ;CLEAR THE MAX ITERATION COUNT
2662 012640 000177 166244 JMP 2$LPERR ;GO TO THE LOOP ADDRESS
2663 012644 032777 040000 166266 3$: BIT #SW14,2SWR ;LOOP ON TEST ?
2664 012652 001054 BNE 6$ ;BR IF LOOPING
2665 012654 104401 017631 TYPE ,CYCLED ;TYPE 'CYCLE DOWN'
2666 012660 104401 017325 TYPE ,SWTCHN ;'SWITCH TO A/B'
2667 012664 104401 017653 TYPE ,CYCLEU ;'CYCLE THE DRIVE UP'
2668 012670 113760 001226 000010 MOVB PORTB,RMCS2(RD) ;SELECT PORT B
2669 012676 013737 001226 001234 MOV PORTB,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2670 012704 032760 010000 000012 4$: BIT #MOL,RMDS1(RD) ;CHECK 'MOL'
2671 012712 001374 BNE 4$ ;BR IF SET (DRIVE NOT CYCLED DOWN)
2672 012714 032760 010000 000012 5$: BIT #MOL,RMDS1(RD) ;CHECK 'MOL' AGAIN
2673 012722 001774 BEQ 5$ ;BR IF NOT SET (DRIVE NOT CYCLED UP)
2674
2675 ;*****
2676 ;SET VOLUME VALID FOR BOTH PORTS
2677
2678 012724 012760 000011 000000 MOV #11,RMCS1(RD) ;ISSUE A DRIVE CLEAR THROUGH PORT B
2679 012732 012760 000021 000000 MOV #21,RMCS1(RD) ;ISSUE A READIN PRESET THROUGH PORT B
2680 012740 012760 000013 000000 MOV #13,RMCS1(RD) ;RELEASE PORT B
2681 012746 113760 001224 000010 MOVB PORTA,RMCS2(RD) ;SELECT PORT A
2682 012754 013737 001224 001234 MOV PORTA,PTNBR ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2683 012762 012760 000021 000000 MOV #21,RMCS1(RD) ;ISSUE A READIN PRESET THROUGH PORT A
2684 012770 012760 010000 000032 MOV #FMT16,RMOF(RD) ;SET FMT16
2685 012776 012760 000013 000000 MOV #13,RMCS1(RD) ;RELEASE PORT A
2686 013004 012737 011610 001254 6$: MOV #5000.,WATCH ;SPINDLE MOTOR 'COOL DOWN' DELAY
2687 013012 005737 001254 7$: TST WATCH ;FINISHED ?
2688 013016 001375 BNE 7$ ;BR IF NOT
2689 013020 000004 SCOPE ;LOOP ?
2690 013022 000137 013026 JMP SEOP ;GO TO THE END OF PASS ROUTINE
2691
2692 .SBTTL END OF PASS ROUTINE
2693
2694 ;*****
2695 ;INCREMENT THE PASS NUMBER ($PASS)
2696 ;INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
2697 ;TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYYY"
2698 ;WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
2699 ;IF THERES A MONITOR GO TO IT
2700 ;IF THERE ISN'T JUMP TO TST1AA
2701
2702 SEOP: 013026 TST KYBCTL ;ENTERED TEST VIA KEYBOARD COMMAND ?
2703 013026 BEQ +6 ;BR IF NOT
2704 013032 001402 JMP EXEC ;RETURN TO KEYBOARD CONTROL
2705 013034 000137 002424 CLRB $TSTNM ;ZERO THE TEST NUMBER
2706 013040 005037 001102 CLR $TIMES ;ZERO THE NUMBER OF ITERATIONS
2707 013044 005037 001176 INC $PASS ;INCREMENT THE PASS NUMBER
2708 013050 005237 001100 BIC #100000,$PASS ;DON'T ALLOW A NEG. NUMBER
2709 013054 042737 100000 001100 DEC (PC)+ ;LOOP?
2710 013062 005327 SEOPCT: .WORD 1 ;YES
2711 013064 000001 BGT $DOAGN ;RESTORE COUNTER
2712 013066 003063 MOV (PC)+,2(PC)+
2713 013070 012737

```


M04

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 51
END OF PASS ROUTINE

SEQ 0051

```

2714 013072 000001
2715 013074 013064
2716 013076 104401 013104
2717 013102 000407
2718
2719 013122
2720 013122 013746 001100
2721
2722 013126 104405
2723 013130 104401 013136
2724 013134 000421
2725
2726 013200
2727 013200 013746 001112
2728
2729 013204 104405
2730 013206 104401 001207
2731 013212 005037 001112
2732 013216 013700 000042
2733 013222 001405
2734 013224 000005
2735 013226 004710
2736 013230 000240
2737 013232 000240
2738 013234 000240
2739 013236
2740 013236 000137
2741 013240 002706
2742 013242 377 377 000
2743 013246
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754 013246 012737 013316 000004
2755 013254 005037 000006
2756 013260 005777 165726
2757 013264 013701 001216
2758 013270 012721 013400
2759 013274 012711 000300
2760 013300 012777 177777 165706
2761 013306 012777 000135 165676
2762 013314 000425
2763 013316 062706 000004
2764 013322 012737 013360 000004
2765 013330 005777 165664
2766 013334 013701 001222
2767 013340 012721 013400
2768 013344 012711 000300
2769 013350 012777 000100 165642

```

```

SENDCT: .WORD 1
          SEOPCT
          TYPE 655
          BR 645
          .ASCIZ <12><15>/END PASS #/
          MOV $PASS,-(SP)
          TYPDS
          TYPE 675
          BR 665
          .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
          MOV $ERTTL,-(SP)
          TYPDS
          TYPE $CRLF
          CLR $ERTTL
          $GET42: MOV $42,R0
                  BEQ $DOAGN
                  RESET
          SENDAD: JSR PC,(R0)
                  NOP
                  NOP
                  NOP
          $DOAGN: JMP $PC)+
                  SRTNAD: .WORD TST1AA
          SENULL: .BYTE -1,-1,0
                  .EVEN

```

```

;; TYPE ASCII STRING
;; GET OVER THE ASCII
;;
;; SAVE $PASS FOR TYPEOUT
;; TYPE PASS NUMBER
;; GO TYPE--DECIMAL ASCII WITH SIGN
;; TYPE ASCII STRING
;; GET OVER THE ASCII
;;
;; SAVE $ERTTL FOR TYPEOUT
;; TOTAL NUMBER OF ERRORS
;; GO TYPE--DECIMAL ASCII WITH SIGN
;; TYPE CARRIAGE RETURN, LINE FEED
;; CLEAR ERROR TOTAL
;; GET MONITOR ADDRESS
;; BRANCH IF NO MONITOR
;; CLEAR THE WORLD
;; GO TO MONITOR
;; SAVE ROOM
;; FOR
;; ACT11

```

```

;; RETURN
;; NULL CHARACTER STRING

```

```

;;*****

```

```

.SBTTL *** SUBROUTINES ***

```

```

;;*****

```

```

;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS

```

```

CKCLK: MOV $CKCLK1,$ERRVEC ;SET UP VECTOR FOR CLOCK CHECK
        CLR $ERRVEC+2 ;NEW PSW
        TST $SLKCSR ;CHECK FOR KW11-P
        MOV $LPVEC,R1 ;KW11-P VECTOR ADDRESS
        MOV $CLOCK,(R1)+ ;SET UP KW11-P VECTOR
        MOV $300,(R1) ;PSW - PRI 6
        MOV $-1,$SLKCSB ;LOAD COUNTER BUFFER WITH 1'S
        MOV $135,$SLKCSR ;SET CLOCK - CNT UP, 16MS, CONT INT
        BR CKCLK3
CKCLK1: ADD $4,SP ;RESTORE THE STACK POINTER
        MOV $CKCLK2,$ERRVEC ;CHANGE ERROR VECTOR TO CHECK FOR KW11-L
        TST $SLKS ;LOOK FOR KW11-L
        MOV $LLVEC,R1 ;KW11-L VECTOR ADDRESS
        MOV $CLOCK,(R1)+ ;SET UP KW11-L VECTOR
        MOV $300,(R1) ;PSW - PRI 6
        MOV $100,$SLKS ;SET KW11-L INTERRUPT

```

```

2770 013356 000404
2771 013360 062706 000004 CKCLK2: BR CKCLK3
2772 013364 062716 000002 ADD #4,SP ;RESTORE THE STACK POINTER
2773 013370 012737 000006 000004 CKCLK3: ADD #2,(SP) ;INCREMENT RETURN, NO CLOCK
2774 013376 000207 MOV #6,#ERRVEC ;RESTORE THE ERROR VECTOR
2775
2776 ;ROUTINE TO COUNT CLOCK TICKS
2777
2778 013400 062737 000021 001252 CLOCK: ADD #17, TIME ;ADD 17 MS TO ELAPSED TIME COUNTER
2779 013406 005737 001254 TST WATCH ;IS WATCH ALREADY ZERO?
2780 013412 001406 BEQ 1$ ;BR IF IT IS
2781 013414 162737 000021 001254 SUB #17, WATCH ;SUBTRACT 17 MS FROM WATCH DOG COUNTER
2782 013422 100002 BPL 1$ ;BR IF NOT MINUS
2783 013424 005037 001254 CLR WATCH ;CLEAR WATCH DOG COUNTER
2784 013430 000002 1$: RTI ;RETURN
2785
2786 ;ROUTINE TO CALCULATE + 25% TIME TOLERANCE VALUES
2787
2788 013432 005746
2789 013434 016616 000002 TOLER: TST -(SP) ;MAKE ROOM ON THE STACK
2790 013440 013546 MOV 2(SP), (SP) ;SAVE STACK
2791 013442 011666 000004 MOV 3(RS), -(SP) ;GET TIME VALUE
2792 013446 006216 MOV (SP), 4(SP) ;MOVE TIME VALUE
2793 013450 006216 ASR (SP) ;DIVIDE BY 2
2794 013452 062666 000002 ASR (SP) ;DIVIDE BY 2 AGAIN (FOR A TOTAL OF 4)
2795 013456 000205 ADD (SP)+, 2(SP) ;CALCULATE UPPER LIMIT FOR TIMEOUT
2796 ;RETURN WITH TOLERANCES ON THE STACK
2797
2798 ;*****
2799 .SBTTL 'SYSMAC' UTILITY ROUTINES
2800
2801 .SBTTL SCOPE HANDLER ROUTINE
2802
2803 ;*****
2804 ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
2805 ;AND LOAD THE TEST NUMBER($TSTN) INTO THE DISPLAY REG.(DISPLAY<7:0>)
2806 ;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
2807 ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2808 ;$SW14=1 LOOP ON TEST
2809 ;$SW11=1 INHIBIT ITERATIONS
2810 ;CALL
2811 ;* SCOPE ;;SCOPE=IOT
2812
2813 $SCOPE:
2814 013460 104407
2815 013462 032777 040000 165450 1$: CKSWR ;TEST FOR CHANGE IN SOFT-SWR
2816 013470 001055 BIT #BIT14, $SWR ;LOOP ON PRESENT TEST?
2817 ;*****START OF CODE FOR THE XOR TESTER***** ;YES IF SW14=1
2818 013472 000416 $XTSTR: BR 6$ ;IF RUNNING ON THE "XOR" TESTER CHANGE
2819 ;THIS INSTRUCTION TO A "NOP" (NOP=240)
2820 013474 013746 000004 MOV 3$ERRVEC, -(SP) ;SAVE THE CONTENTS OF THE ERROR VECTOR
2821 013500 012737 013520 000004 MOV 5$, 3$ERRVEC ;SET FOR TIMEOUT
2822 013506 005737 177060 TST 3$177060 ;TIME OUT ON XOR?
2823 013512 012637 000004 MOV (SP)+, 3$ERRVEC ;RESTORE THE ERROR VECTOR
2824 013516 000436 BR 5$VLAD ;GO TO THE NEXT TEST
2825 013520 022626 5$: CMP (SP)+, (SP)+ ;CLEAR THE STACK AFTER A TIME OUT

```



```

2826 013522 012637 000004      MOV      (SP)+, @ERRVEC      ;; RESTORE THE ERROR VECTOR
2827 013526 000436      BR      SOVER          ;; LOOP ON THE PRESENT TEST
2828 013530      6$:; ##### END OF CODE FOR THE XOR TESTER #####
2829 013530 105737 001103      2$: TSTB     SERFLG      ;; HAS AN ERROR OCCURRED?
2830 013534 001404      BEQ      3$          ;; BR IF NO
2831 013536 105037 001103      4$: CLRB     SERFLG      ;; ZERO THE ERROR FLAG
2832 013542 005037 001176      CLR      STIMES      ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
2833 013546 032777 004000 165364 3$: BIT      @BIT11, @SWR      ;; INHIBIT ITERATIONS?
2834 013554 001011      BNE      1$          ;; BR IF YES
2835 013556 005737 001100      TST      SPASS      ;; IF FIRST PASS OF PROGRAM
2836 013562 001406      BEQ      1$          ;; INHIBIT ITERATIONS
2837 013564 005237 001104      INC      $ICNT      ;; INCREMENT ITERATION COUNT
2838 013570 023737 001176 001104      CMP      STIMES, $ICNT      ;; CHECK THE NUMBER OF ITERATIONS MADE
2839 013576 002012      BGE      SOVER      ;; BR IF MORE ITERATION REQUIRED
2840 013600 012737 000001 001104 1$: MOV      #1, $ICNT      ;; REINITIALIZE THE ITERATION COUNTER
2841 013606 013737 013640 001176      MOV      $MXCNT, STIMES      ;; SET NUMBER OF ITERATIONS TO DO
2842 013614 105237 001102      $SVLAD: INCB     $TSTNM      ;; COUNT TEST NUMBERS
2843 013620 011637 001106      MOV      (SP), $LPADR      ;; SAVE SCOPE LOOP ADDRESS
2844 013624 013777 001102 165310      SOVER: MOV      $TSTNM, @DISPLAY      ;; DISPLAY TEST NUMBER
2845 013632 013716 001106      MOV      $LPADR, (SP)      ;; FUDGE RETURN ADDRESS
2846 013636 000002      RTI          ;; FIXES PS
2847 013640 000004      $MXCNT: 4          ;; MAX. NUMBER OF ITERATIONS
2848      .SBTTL  ERROR HANDLER ROUTINE
2849
2850      ;; *****
2851      ;; THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2852      ;; SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2853      ;; AND GO TO SERRTYP ON ERROR
2854      ;; THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2855      ;; $SW15=1      HALT ON ERROR
2856      ;; $SW13=1      INHIBIT ERROR TYPEOUTS
2857      ;; $SW10=1      BELL ON ERROR
2858      ;; $CALL
2859      ;; *      ERROR      N      ;; ERROR=EMT AND N=ERROR ITEM NUMBER
2860
2861 013642      SERROR:      CKSWR      ;; TEST FOR CHANGE IN SOFT-SWR
2862 013642 104407 001102 001242      MOV      $TSTNM, $TSTNM      ;; SET THE ERROR FLAG
2863 013644 113737 001103      7$: INCB     SERFLG      ;; DON'T LET THE FLAG GO TO ZERO
2864 013652 105237 001103      BEQ      7$          ;; DISPLAY TEST NUMBER AND ERROR FLAG
2865 013656 001775      MOV      $TSTNM, @DISPLAY      ;; BELL ON ERROR?
2866 013660 013777 001102 165254      BIT      @BIT10, @SWR      ;; NO - SKIP
2867 013666 032777 002000 165244      BEQ      1$          ;; RING BELL
2868 013674 001402      TYPE     $BELL      ;; COUNT THE NUMBER OF ERRORS
2869 013676 104401 001202      1$: INC      $ERTTL      ;; GET ADDRESS OF ERROR INSTRUCTION
2870 013702 005237 001112      MOV      (SP), $ERRPC      ;; STRIP AND SAVE THE ERROR ITEM CODE
2871 013706 011637 001116      SUB      #2, $ERRPC      ;; SKIP TYPEOUT IF SET
2872 013712 162737 000002 001116      MOV      @ERRPC, $ITEMB      ;; SKIP TYPEOUTS
2873 013720 117737 165172 001114      BIT      @BIT13, @SWR      ;; GO TO USER ERROR ROUTINE
2874 013726 032777 020000 165204      BNE      20$          ;;
2875 013734 001004      JSR      PC, SERRTYP      ;;
2876 013736 004737 013774      TYPE     $CRLF      ;;
2877 013742 104401 001207      20$:      TST      @SWR      ;; HALT ON ERROR
2878 013746      2$:      BPL      3$          ;; SKIP IF CONTINUE
2879 013746 005777 165166      BPL      3$          ;; HALT ON ERROR!
2880 013752 100002
2881 013754 000000      HALT

```

```

2882 013756 104407
2883 013760
2884 013760 022737 013226 000042
2885 013766 001001
2886 013770 000000
2887 013772
2888 013772 000002
2889
2890
2891
2892
2893
2894
2895
2896 013774
2897 013774 104401 001207
2898 014000 010046
2899 014002 005000
2900 014004 153700 001114
2901 014010 001004
2902
2903 014012 013746 001116
2904
2905 014016 104402
2906 014020 000445
2907 014022 005300
2908 014024 006300
2909 014026 006300
2910 014030 006300
2911 014032 062700 001276
2912 014036 012037 014046
2913 014042 001404
2914 014044 104401
2915 014046 000000
2916 014050 104401 001207
2917 014054 012037 014064
2918 014060 001404
2919 014062 104401
2920 014064 000000
2921 014066 104401 001207
2922 014072 010146
2923 014074 012001
2924 014076 001415
2925 014100 012000
2926 014102 105720
2927 014104 001003
2928 014106 013146
2929 014110 104402
2930 014112 000402
2931 014114
2932 014114 013146
2933 014116 104405
2934 014120 005711
2935 014122 001403
2936 014124 104401 014144
2937 014130 000764

          CKSWR                      ;;TEST FOR CHANGE IN SOFT-SWR
3$:      CMP      #SENDAD,2#42      ;;ACT-11 AUTO-ACCEPT?
          BNE      6$              ;;BRANCH IF NO
          HALT                      ;;YES
6$:      RTI                        ;;RETURN
.SBTTL   ERROR MESSAGE TYPEOUT ROUTINE

*****
;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
;ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
;AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
SERRTYP:
          TYPE      $CRLF            ;; "CARRIAGE RETURN" & "LINE FEED"
          MOV      R0,-(SP)          ;;SAVE R0
          CLR      R0                ;;PICKUP THE ITEM INDEX
          BISB     2#$ITEMB,R0
          BNE      1$
          MOV      $ERRPC,-(SP)      ;; IF ITEM NUMBER IS ZERO, JUST
          ;;TYPE THE PC OF THE ERROR
          ;;SAVE $ERRPC FOR TYPEOUT
          ;;ERROR ADDRESS
          GO TYPE--OCTAL ASCII(ALL DIGITS)
          GET OUT
          ADJUST THE INDEX SO THAT IT WILL
          ;;WORK FOR THE ERROR TABLE
1$:      DEC      R0
          ASL      R0
          ASL      R0
          ASL      R0
          ADD      #SERRTB,R0
          MOV      (R0)+,2$
          BEQ      3$
          TYPE      0
          WORD      0
          TYPE      $CRLF
          MOV      (R0)+,4$
          BEQ      5$
          TYPE      0
          WORD      0
          TYPE      $CRLF
          MOV      R1,-(SP)
          MOV      (R0)+,R1
          BEQ      9$
          MOV      (R0)+,R0
          TSTB     (R0)+
          BNE      7$
          MOV      2(R1)+,-(SP)
          TPOC
          BR       8$
          BR       8$
          MOV      2(R1)+,-(SP)
          TYPDS
          TST      (R1)
          BEQ      9$
          TYPE      11$
          BR       6$
          ;;FORM TABLE POINTER
          ;;PICKUP "ERROR MESSAGE" POINTER
          ;;SKIP TYPEOUT IF NO POINTER
          ;;TYPE THE "ERROR MESSAGE"
          ;; "ERROR MESSAGE" POINTER GOES HERE
          ;; "CARRIAGE RETURN" & "LINE FEED"
          ;;PICKUP "DATA HEADER" POINTER
          ;;SKIP TYPEOUT IF 0
          ;;TYPE THE "DATA HEADER"
          ;; "DATA HEADER" POINTER GOES HERE
          ;; "CARRIAGE RETURN" & "LINE FEED"
          ;;SAVE R1
          ;;PICKUP "DATA TABLE" POINTER
          ;;BR IF NO DATA TO BE TYPED
          ;;PICKUP "DATA FORMAT" POINTER
          ;; "OCTAL" OR "DECIMAL"
          ;;BR IF DECIMAL
          ;;SAVE 2(R1)+ FOR TYPEOUT
          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
          ;;SAVE 2(R1)+ FOR TYPEOUT
          ;;GO TYPE--DECIMAL ASCII WITH SIGN
          ;;IS THERE ANOTHER NUMBER?
          ;;BR IF NO
          ;;TYPE TWO(2) SPACES
          ;;LOOP

```


2938
2939 014132 012601
2940 014134 012600
2941 014136 104401 001207
2942 014142 000207
2943 014144 020040 000
2944 014150
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962 014150 105737 001157
2963 014154 100002
2964 014156 000000
2965 014160 000407
2966 014162 010046
2967 014164 017600 000002
2968 014170 112046
2969 014172 001005
2970 014174 005726
2971 014176 012600
2972 014200 062716 000002
2973 014204 000002
2974 014206 122716 000011
2975 014212 001430
2976 014214 122716 000200
2977 014220 001006
2978 014222 005726
2979 014224 104401
2980 014226 001207
2981 014230 105037 014364
2982 014234 000755
2983 014236 004737 014320
2984 014242 123726 001156
2985 014246 001350
2986 014250 013746 001154
2987
2988 014254 105366 000001
2989 014260 002770
2990 014262 004737 014320
2991 014266 105337 014364
2992 014272 000770
2993

```

9$:      MOV      (SP)+,R1      ;;RESTORE R1
10$:     MOV      (SP)+,R0      ;;RESTORE R0
        TYPE      $CRLF        ;; "CARRIAGE RETURN" & "LINE FEED"
        RTS       PC           ;; RETURN
11$:     .ASCIZ   / /          ;; TWO(2) SPACES
        .EVEN
        .SBTTL   TYPE ROUTINE

*****
; ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
; THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
; NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
; NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
; NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
;
; CALL:
; 1) USING A TRAP INSTRUCTION
;      TYPE      ,MESADR      ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
; OR
;      TYPE
;      MESADR
;
$TYPE:   TSTB     $TPFLG        ;; IS THERE A TERMINAL?
        BPL      1$            ;; BR IF YES
        HALT     IF NO TERMINAL
        LEAVE
1$:      MOV      R0, -(SP)      ;; SAVE R0
        MOV      @2(SP), R0     ;; GET ADDRESS OF ASCIZ STRING
2$:      MOVB     (R0)+, -(SP)   ;; PUSH CHARACTER TO BE TYPED ONTO STACK
        BNE      4$            ;; BR IF IT ISN'T THE TERMINATOR
        TST      (SP)+          ;; IF TERMINATOR POP IT OFF THE STACK
60$:     MOV      (SP)+, R0      ;; RESTORE R0
3$:      ADD      #2, (SP)       ;; ADJUST RETURN PC
        RTI
4$:      CMPB     #HT, (SP)      ;; BRANCH IF <HT>
        BEQ      8$
        CMPB     #CRLF, (SP)    ;; BRANCH IF NOT <CRLF>
        BNE      5$
        TST      (SP)+          ;; POP <CR><LF> EQUIV
        TYPE     A CR AND LF
        CLEAR    CHARACTER COUNT
        GET NEXT CHARACTER
        GO TYPE THIS CHARACTER
5$:      JSR      PC, $TYPEPC    ;; IS IT TIME FOR FILLER CHARS.?
6$:      CMPB     $FILLC, (SP)+  ;; IF NO GO GET NEXT CHAR.
        BNE      2$
        MOV      $NULL, -(SP)   ;; GET # OF FILLER CHARS. NEEDED
        AND THE NULL CHAR.
        DOES A NULL NEED TO BE TYPED?
        BR IF NO--GO POP THE NULL OFF OF STACK
        GO TYPE A NULL
        DO NOT COUNT AS A COUNT
7$:      DECB     1(SP)
        BLT      6$
        JSR      PC, $TYPEPC
        DECB     $CHARCNT
        BR       7$
        LOOP

```

```

2994 ;HORIZONTAL TAB PROCESSOR
2995
2996 014274 112716 000040 8S: MOVB #' (SP) ;: REPLACE TAB WITH SPACE
2997 014300 004737 014320 9S: JSR PC, $TYPEPC ;: TYPE A SPACE
2998 014304 132737 000007 014364 BITB #7, $CHARCNT ;: BRANCH IF NOT AT
2999 014312 001372 BNE 9S ;: TAB STOP
3000 014314 005726 TST (SP)+ ;: POP SPACE OFF STACK
3001 014316 000724 BR 2S ;: GET NEXT CHARACTER
3002 014320 105777 164624 $TYPEPC: TSTB $STPS ;: WAIT UNTIL PRINTER IS READY
3003 014324 100375 BPL $TYPEPC
3004 014326 116677 000002 164616 MOVB 2(SP), $STPB ;: LOAD CHAR TO BE TYPED INTO DATA REG.
3005 014334 122766 000015 000002 CMPB #CR, 2(SP) ;: IS CHARACTER A CARRIAGE RETURN?
3006 014342 001003 BNE 1S ;: BRANCH IF NO
3007 014344 105037 014364 CLRB $CHARCNT ;: YES--CLEAR CHARACTER COUNT
3008 014350 000406 BR $TYPEX ;: EXIT
3009 014352 122766 000012 000002 1S: CMPB #LF, 2(SP) ;: IS CHARACTER A LINE FEED?
3010 014360 001402 BEQ $TYPEX ;: BRANCH IF YES
3011 014362 105227 INCB (PC)+ ;: COUNT THE CHARACTER
3012 014364 000000 $CHARCNT: .WORD 0 ;: CHARACTER COUNT STORAGE
3013 014366 000207 $TYPEX: RTS PC
3014
3015 .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
3016
3017 ;: *****
3018 ;: THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3019 ;: OCTAL (ASCII) NUMBER AND TYPE IT.
3020 ;: $TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3021 ;: CALL:
3022 ;: MOV NUM, -(SP) ;: NUMBER TO BE TYPED
3023 ;: TYPOS ;: CALL FOR TYPEOUT
3024 ;: .BYTE N ;: N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3025 ;: .BYTE M ;: M=1 OR 0
3026 ;: ;: 1=TYPE LEADING ZEROS
3027 ;: ;: 0=SUPPRESS LEADING ZEROS
3028
3029 ;: $STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3030 ;: $TYPOS OR $TYPOC
3031 ;: CALL:
3032 ;: MOV NUM, -(SP) ;: NUMBER TO BE TYPED
3033 ;: TYPON ;: CALL FOR TYPEOUT
3034
3035 ;: $TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3036 ;: CALL:
3037 ;: MOV NUM, -(SP) ;: NUMBER TO BE TYPED
3038 ;: TYPOC ;: CALL FOR TYPEOUT
3039
3040 014370 017646 000000 014613 $TYPOS: MOV 2(SP), -(SP) ;: PICKUP THE MODE
3041 014374 116637 000001 MOVB 1(SP), $OFILL ;: LOAD ZERO FILL SWITCH
3042 014402 112637 014615 MOVB (SP)+, $OMODE+1 ;: NUMBER OF DIGITS TO TYPE
3043 014406 062716 000002 ADD #2, (SP) ;: ADJUST RETURN ADDRESS
3044 014412 000406 BR $STYPON
3045 014414 112737 000001 014613 $TYPOC: MOVB #1, $OFILL ;: SET THE ZERO FILL SWITCH
3046 014422 112737 000006 014615 MOVB #6, $OMODE+1 ;: SET FOR SIX(6) DIGITS
3047 014430 112737 000005 014612 $STYPON: MOVB #5, $OCNT ;: SET THE ITERATION COUNT
3048 014436 010346 MOV R3, -(SP) ;: SAVE R3
3049 014440 010446 MOV R4, -(SP) ;: SAVE R4

```


3050	014442	010546		MOV	R5, -(SP)	::SAVE R5
3051	014444	113704	014615	MOVB	\$OMODE+1, R4	::GET THE NUMBER OF DIGITS TO TYPE
3052	014450	005404		NEG	R4	
3053	014452	062704	000006	ADD	#6, R4	::SUBTRACT IT FOR MAX. ALLOWED
3054	014456	110437	014614	MOVB	R4, \$OMODE	::SAVE IT FOR USE
3055	014462	113704	014613	MOVB	\$OFILL, R4	::GET THE ZERO FILL SWITCH
3056	014466	016605	000012	MOV	12(SP), R5	::PICKUP THE INPUT NUMBER
3057	014472	005003		CLR	R3	::CLEAR THE OUTPUT WORD
3058	014474	006105		1\$: ROL	R5	::ROTATE MSB INTO "C"
3059	014476	000404		BR	3\$	::GO DO MSB
3060	014500	006105		2\$: ROL	R5	::FORM THIS DIGIT
3061	014502	006105		ROL	R5	
3062	014504	006105		ROL	R5	
3063	014506	010503		MOV	R5, R3	
3064	014510	006103		3\$: ROL	R3	::GET LSB OF THIS DIGIT
3065	014512	105337	014614	DECB	\$OMODE	::TYPE THIS DIGIT?
3066	014516	100016		BPL	7\$	::BR IF NO
3067	014520	042703	177770	BIC	#177770, R3	::GET RID OF JUNK
3068	014524	001002		BNE	4\$	::TEST FOR 0
3069	014526	005704		TST	R4	::SUPPRESS THIS 0?
3070	014530	001403		BEQ	5\$	::BR IF YES
3071	014532	005204		4\$: INC	R4	::DON'T SUPPRESS ANYMORE 0'S
3072	014534	052703	000060	BIS	#'0, R3	::MAKE THIS DIGIT ASCII
3073	014540	052703	000040	5\$: BIS	#' , R3	::MAKE ASCII IF NOT ALREADY
3074	014544	110337	014610	MOVB	R3, 8\$	::SAVE FOR TYPING
3075	014550	104401	014610	TYPE	8\$	::GO TYPE THIS DIGIT
3076	014554	105337	014612	7\$: DECB	\$OCNT	::COUNT BY 1
3077	014560	003347		BGT	2\$	::BR IF MORE TO DO
3078	014562	002402		BLT	6\$	::BR IF DONE
3079	014564	005204		INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
3080	014566	000744		BR	2\$	::GO DO THE LAST DIGIT
3081	014570	012605		6\$: MOV	(SP)+, R5	::RESTORE R5
3082	014572	012604		MOV	(SP)+, R4	::RESTORE R4
3083	014574	012603		MOV	(SP)+, R3	::RESTORE R3
3084	014576	016666	000002 000004	MOV	2(SP), 4(SP)	::SET THE STACK FOR RETURNING
3085	014604	012616		MOV	(SP)+, (SP)	
3086	014606	000002		RTI		::RETURN
3087	014610	000		8\$: .BYTE	0	::STORAGE FOR ASCII DIGIT
3088	014611	000		.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
3089	014612	000		\$OCNT: .BYTE	0	::OCTAL DIGIT COUNTER
3090	014613	000		\$OFILL: .BYTE	0	::ZERO FILL SWITCH
3091	014614	000000		\$OMODE: .WORD	0	::NUMBER OF DIGITS TO TYPE
3092				.SBTTL	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE	
3093						
3094						
3095						
3096						
3097						
3098						
3099						
3100						
3101						
3102						
3103						
3104	014616					
3105	014616	010046		\$TYPDS: MOV	RO, -(SP)	::PUSH RO ON STACK

::*****
 ::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
 ::*

	014620	010146		MOV	R1,-(SP)	:: PUSH R1 ON STACK
3107	014622	010246		MOV	R2,-(SP)	: PUSH R2 ON STACK
3108	014624	010346		MOV	R3,-(SP)	: PUSH R3 ON STACK
3109	014626	010546		MOV	R5,-(SP)	: PUSH R5 ON STACK
3110	014630	012746	020200	MOV	#20200,-(SP)	: SET BLANK SWITCH AND SIGN
3111	014634	016605	000020	MOV	20(SP),R5	: GET THE INPUT NUMBER
3112	014640	100004		BPL	1\$	: BR IF INPUT IS POS.
3113	014642	005405		NEG	R5	: MAKE THE BINARY NUMBER POS.
3114	014644	112766	000055 000001	MOVB	#'-,1(SP)	: MAKE THE ASCII NUMBER NEG.
3115	014652	005000		CLR	R0	: ZERO THE CONSTANTS INDEX
3116	014654	012703	015032	MOV	\$DBLK,R3	: SETUP THE OUTPUT POINTER
3117	014660	112723	000040	MOVB	#',(R3)+	: SET THE FIRST CHARACTER TO A BLANK
3118	014664	005002		CLR	R2	: CLEAR THE BCD NUMBER
3119	014666	016001	015022	MOV	\$DTBL(R0),R1	: GET THE CONSTANT
3120	014672	160105		SUB	R1,R5	: FORM THIS BCD DIGIT
3121	014674	002402		BLT	4\$	: BR IF DONE
3122	014676	005202		INC	R2	: INCREASE THE BCD DIGIT BY 1
3123	014700	000774		BR	3\$	:
3124	014702	060105		ADD	R1,R5	: ADD BACK THE CONSTANT
3125	014704	005702		TST	R2	: CHECK IF BCD DIGIT=0
3126	014706	001002		BNE	5\$	: FALL THROUGH IF 0
3127	014710	105716		TSTB	(SP)	: STILL DOING LEADING O'S?
3128	014712	100407		BMI	7\$	: BR IF YES
3129	014714	106316		ASLB	(SP)	: MSD?
3130	014716	103003		BCC	6\$	: BR IF NO
3131	014720	116663	000001 177777	MOVB	1(SP),-1(R3)	: YES--SET THE SIGN
3132	014726	052702	000060	BIS	#'O,R2	: MAKE THE BCD DIGIT ASCII
3133	014732	052702	000040	BIS	#',R2	: MAKE IT A SPACE IF NOT ALREADY A DIGIT
3134	014736	110223		MOVB	R2,(R3)+	: PUT THIS CHARACTER IN THE OUTPUT BUFFER
3135	014740	005720		TST	(R0)+	: JUST INCREMENTING
3136	014742	020027	000010	CMP	R0,#10	: CHECK THE TABLE INDEX
3137	014746	002746		BLT	2\$	: GO DO THE NEXT DIGIT
3138	014750	003002		BGT	8\$	: GO TO EXIT
3139	014752	010502		MOV	R5,R2	: GET THE LSD
3140	014754	000764		BR	6\$	: GO CHANGE TO ASCII
3141	014756	105726		TSTB	(SP)+	: WAS THE LSD THE FIRST NON-ZERO?
3142	014760	100003		BPL	9\$	: BR IF NO
3143	014762	116663	177777 177776	MOVB	-1(SP),-2(R3)	: YES--SET THE SIGN FOR TYPING
3144	014770	105013		CLRB	(R3)	: SET THE TERMINATOR
3145	014772	012605		MOV	(SP)+,R5	: POP STACK INTO R5
3146	014774	012603	.	MOV	(SP)+,R3	: POP STACK INTO R3
3147	014776	012602		MOV	(SP)+,R2	: POP STACK INTO R2
3148	015000	012601		MOV	(SP)+,R1	: POP STACK INTO R1
3149	015002	012600		MOV	(SP)+,R0	: POP STACK INTO R0
3150	015004	104401	015032	TYPE	\$DBLK	: NOW TYPE THE NUMBER
3151	015010	016666	000002 000004	MOV	2(SP),4(SP)	: ADJUST THE STACK
3152	015016	012616		MOV	(SP)+,(SP)	:
3153	015020	000002		RTI		: RETURN TO USER
3154	015022	023420		\$DTBL:	10000.	
3155	015024	001750			1000.	
3156	015026	000144			100.	
3157	015030	000012			10.	
3158	015032	000004		\$DBLK:	.BLKW 4	
3159				.SBTTL	TTY INPUT ROUTINE	
3160						
3161						::*****


```
3162      .ENABL  LSB
3163      STKCNT: .WORD  0          ;; NUMBER OF ITEMS IN QUEUE
3164      STKQIN:  .WORD  0          ;; INPUT POINTER
3165      STKQOUT: .WORD  0          ;; OUTPUT POINTER
3166      STKQSR:  .BLKB  1          ;; TTY KEYBOARD QUEUE
3167      STKQEND=.
3168      .EVEN
3169
3170      ;*TK INITIALIZE ROUTINE
3171      ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
3172      ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
3173
3174      ;*CALL:
3175      ;*      JSR      PC,STKINT
3176      ;*      RETURN
3177
3178      STKINT: CLR      STKCNT          ;; CLEAR COUNT OF ITEMS IN QUEUE
3179              MOV      STKQSR,STKQIN  ;; MOVE THE STARTING ADDRESS OF THE
3180              MOV      STKQIN,STKQOUT ;; QUEUE INTO THE INPUT & OUTPUT POINTERS.
3181              MOV      STKSRV,STKVEC  ;; INITIALIZE THE KEYBOARD VECTOR
3182              MOV      #200,STKVEC+2  ;; "BR" LEVEL 4
3183              TST      STKB           ;; CLEAR DONE FLAG
3184              MOV      #100,STKS      ;; ENABLE TTY KEYBOARD INTERRUPT
3185              RTS      PC             ;; RETURN TO CALLER
3186
3187      ;*TK SERVICE ROUTINE
3188      ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
3189      ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
3190      ;*IT IN THE QUEUE.
3191
3192      STKSRV: MOVB     STKB,-(SP)      ;; PICKUP THE CHARACTER
3193              BIC      #177,(SP)      ;; STRIP THE JUNK
3194              CMP      (SP),#7        ;; IS IT A CONTROL G?
3195              BNE      2$             ;; BRANCH IF NO
3196              CMP      #SWREG,SWR     ;; IS SOFT-SWR SELECTED?
3197              BEQ      6$             ;; GO TO SWR CHANGE
3198
3199      2$:      CMP      #1,STKCNT      ;; IS THE QUEUE FULL?
3200              BNE      3$             ;; BRANCH IF NO
3201              TYPE     $BELL          ;; RING THE TTY BELL
3202              TST      (SP)+          ;; CLEAN CHARACTER OFF OF STACK
3203              BR       5$             ;; EXIT
3204      3$:      CMP      (SP),#23      ;; IS IT A CONTROL-S?
3205              BNE      32$           ;; BRANCH IF NO
3206              CLR      STKS          ;; DISABLE TTY KEYBOARD INTERRUPTS
3207              TST      (SP)+          ;; CLEAN CHAR OFF STACK
3208              TSTB     STKS          ;; WAIT FOR A CHAR
3209              BPL      31$           ;; LOOP UNTIL ITS THERE
3210              MOVB     STKB,-(SP)    ;; GET THE CHARACTER
3211              BIC      #177,(SP)    ;; MAKE IT 7-BIT ASCII
3212              CMP      (SP)+,#21    ;; IS IT A CONTROL-Q?
3213              BNE      31$          ;; BRANCH IF NO
3214              MOV      #100,STKS    ;; REENABLE TTY KEYBOARD INTERRUPTS
3215              RTI
3216      31$:     INC      STKCNT        ;; COUNT THIS CHARACTER
3217      32$:     INC      STKCNT
```

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 60
TTY INPUT ROUTINE

SEQ 0060

```

3218 015244 021627 000140      CMP      (SP),#140      ;; IS IT UPPER CASE?
3219 015250 002405      BLT      4$      ;; BRANCH IF YES
3220 015252 021627 000175      CMP      (SP),#175      ;; IS IT A SPECIAL CHAR?
3221 015256 003002      BGT      4$      ;; BRANCH IF YES
3222 015260 042716 000040      BIC      #40,(SP)      ;; MAKE IT UPPER CASE
3223 015264 112677 177554      4$: MOV     (SP)+,@STKQIN  ;; AND PUT IT IN QUEUE
3224 015270 005237 015044      INC      STKQIN      ;; UPDATE THE POINTER
3225 015274 023727 015044 015051  CMP     STKQIN,@STKQEND  ;; GO OFF THE END?
3226 015302 001003      BNE      5$      ;; BRANCH IF NO
3227 015304 012737 015050 015044  MOV     @STKQSR,STKQIN  ;; RESET THE POINTER
3228 015312 000002      5$: RTI      ;; RETURN
3229
3230      ;; *****
3231      ;; SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
3232      ;; ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
3233      ;; SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
3234      ;; CALL WHEN OPERATING IN TTY INTERRUPT MODE.
3235 015314 022737 000176 001140  $CKSWR: CMP     @SWREG,SWR      ;; IS THE SOFT-SWR SELECTED
3236 015322 001104      BNE      15$      ;; EXIT IF NOT
3237 015324 105777 163614      TSTB     @STKS      ;; IS A CHAR WAITING?
3238 015330 100101      BPL      15$      ;; IF NOT, EXIT
3239 015332 117746 163610      MOV     @STKB,-(SP)      ;; YES
3240 015336 042716 177600      BIC      #177,(SP)      ;; MAKE IT 7-BIT ASCII
3241 015342 021627 000007      CMP      (SP),#7      ;; IS IT A CONTROL-G?
3242 015346 001300      BNE      2$      ;; IF NOT, PUT IT IN THE TTY QUEUE
3243      ;; AND EXIT
3244
3245      ;; *****
3246      ;; CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
3247      ;; ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
3248      ;; CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
3249 015350 123727 001134 000001  6$: CMP     @AUTOB,#1      ;; ARE WE RUNNING IN AUTO-MODE?
3250 015356 001674      BEQ      2$      ;; BRANCH IF YES
3251 015360 005726      TST      (SP)+      ;; CLEAR CONTROL-G OFF STACK
3252 015362 004737 015052      JSR      PC,STKINT      ;; FLUSH THE TTY INPUT QUEUE
3253 015366 005077 163552      CLR      @STKS      ;; DISABLE TTY KEYBOARD INTERRUPTS
3254 015372 112737 000001 001135  MOV     #1,$INTAG      ;; SET INTERRUPT MODE INDICATOR
3255
3256 015400 104401 016156      TYPE     ,SCNTLG      ;; ECHO THE CONTROL-G (↑G)
3257 015404 104401 016163      SGTSWR: TYPE     ,SMSWR      ;; TYPE CURRENT CONTENTS
3258 015410 013746 000176      MOV     SWREG,-(SP)      ;; SAVE SWREG FOR TYPEOUT
3259 015414 104402      TYPOC      ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
3260 015416 104401 016174      TYPE     ,SMNEW      ;; PROMPT FOR NEW SWR
3261 015422 005046      19$: CLR     -(SP)      ;; CLEAR COUNTER
3262 015424 005046      CLR      -(SP)      ;; THE NEW SWR
3263 015426 105777 163512      7$: TSTB     @STKS      ;; CHAR THERE?
3264 015432 100375      BPL      7$      ;; IF NOT TRY AGAIN
3265
3266 015434 117746 163506      MOV     @STKB,-(SP)      ;; PICK UP CHAR
3267 015440 042716 177600      BIC      #177,(SP)      ;; MAKE IT 7-BIT ASCII
3268
3269
3270
3271 015444 021627 000025      9$: CMP     (SP),#25      ;; IS IT A CONTROL-U?
3272 015450 001005      BNE      10$      ;; BRANCH IF NOT
3273 015452 104401 016151      TYPE     ,SCNTLU      ;; YES, ECHO CONTROL-U (↑U)

```



```

3274 015456 062706 000006      20$: ADD    #6,SP      ;; IGNORE PREVIOUS INPUT
3275 015462 000757              BR      19$      ;; LET'S TRY IT AGAIN
3276
3277
3278 015464 021627 000015      10$: CMP      (SP),#15      ;; IS IT A <CR>?
3279 015470 001022              BNE      16$      ;; BRANCH IF NO
3280 015472 005766 000004              TST      4(SP)      ;; YES, IS IT THE FIRST CHAR?
3281 015476 001403              BEQ      11$      ;; BRANCH IF YES
3282 015500 016677 000002 163432      MOV      2(SP),@SWR      ;; SAVE NEW SWR
3283 015506 062706 000006      ADD      #6,SP      ;; CLEAR UP STACK
3284 015512 104401 001207      11$: TYPE    $CRLF      ;; ECHO <CR> AND <LF>
3285 015516 123727 001135 000001      14$: CMPB    $INTAG,#1      ;; RE-ENABLE TTY KBD INTERRUPTS?
3286 015524 001003              BNE      15$      ;; BRANCH IF NOT
3287 015526 012777 000100 163410      MOV      #100,@STKS      ;; RE-ENABLE TTY KBD INTERRUPTS
3288 015534 000002              RTI              ;; RETURN
3289 015536 004737 014320      15$: JSR      PC,$TYPEC      ;; ECHO CHAR
3290 015542 021627 000060      16$: CMP      (SP),#60      ;; CHAR < 0?
3291 015546 002420              BLT      18$      ;; BRANCH IF YES
3292 015550 021627 000067              CMP      (SP),#67      ;; CHAR > 7?
3293 015554 003015              BGT      18$      ;; BRANCH IF YES
3294 015556 042726 000060              BIC      #60,(SP)+      ;; STRIP-OFF ASCII
3295 015562 005766 000002              TST      2(SP)      ;; IS THIS THE FIRST CHAR
3296 015566 001403              BEQ      17$      ;; BRANCH IF YES
3297 015570 006316              ASL      (SP)      ;; NO, SHIFT PRESENT
3298 015572 006316              ASL      (SP)      ;; CHAR OVER TO MAKE
3299 015574 006316              ASL      (SP)      ;; ROOM FOR NEW ONE.
3300 015576 005266 000002      17$: INC      2(SP)      ;; KEEP COUNT OF CHAR
3301 015602 056616 177776              BIS      -2(SP),(SP)      ;; SET IN NEW CHAR
3302 015606 000707              BR      7$      ;; GET THE NEXT ONE
3303 015610 104401 001206      18$: TYPE    $QUES      ;; TYPE ?<CR><LF>
3304 015614 000720              BR      20$      ;; SIMULATE CONTROL-U
3305 .DSABL  LSB
3306
3307
3308 *****
3309 *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
3310 *CALL:
3311 *      RDCHR
3312 *      RETURN HERE
3313 *
3314 *
3315 *
3316 015616 011646 000004 000002 $RDCHR: MOV      (SP),-(SP)      ;; PUSH DOWN THE PC AND
3317 015620 016666 000004              MOV      4(SP),2(SP)      ;; THE PS
3318 015626 005066 000004              CLR      4(SP)      ;; GET READY FOR A CHARACTER
3319 015632 005046              CLR      -(SP)      ;; PUT NEW PS ON STACK
3320 015634 012746 015642              MOV      #64$,-(SP)      ;; PUT NEW PC ON STACK
3321 015640 000002              RTI              ;; POP NEW PC AND PS
3322 015642
3323 015642 005737 015042      64$: TST      $TKCNT      ;; WAIT ON A CHARACTER
3324 015646 001775      1$: BEQ      1$
3325 015650 005337 015042              DEC      $TKCNT      ;; DECREMENT THE COUNTER
3326 015654 117766 177166 000004      MOVB    @STKGOUT,4(SP)      ;; GET ONE CHARACTER
3327 015662 005237 015046              INC      $TKGOUT      ;; UPDATE THE POINTER
3328 015666 023727 015046 015051      CMP      $TKGOUT,$TKGEND      ;; DID IT GO OFF OF THE END?
3329 015674 001003              BNE      2$      ;; BRANCH IF NO

```

```

3330 015676 012737 015050 015046      MOV      #STKQSR,STKQOUT ;:RESET THE POINTER
3331 015704 000002                      RTI          ;:RETURN
3332                                     ;:*****
3333                                     ;:THIS ROUTINE WILL INPUT A STRING FROM THE TTY
3334                                     ;:CALL:
3335                                     ;:
3336                                     ;:RD LIN          ;:INPUT A STRING FROM THE TTY
3337                                     ;:RETURN HERE      ;:ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
3338                                     ;:                ;:TERMINATOR WILL BE A BYTE OF ALL 0'S
3339 015706 010346      $RD LIN: MOV      R3, -(SP)      ;:SAVE R3
3340 015710 005046      CLR          -(SP)              ;:CLEAR THE RUBOUT KEY
3341 015712 012703 016142 1$:      MOV      #TTYIN, R3    ;:GET ADDRESS
3342 015716 022703 016151 2$:      CMP      #TTYIN+7, R3  ;:BUFFER FULL?
3343 015722 101456      BLOS          4$                  ;:BR IF YES
3344 015724 104410      RDCHR          ;:GO READ ONE CHARACTER FROM THE TTY
3345 015726 112613      MOVB          (SP)+, (R3)        ;:GET CHARACTER
3346 015730 122713 000177 10$:     CMPB      #177, (R3)   ;:IS IT A RUBOUT
3347 015734 001022      BNE          5$                  ;:BR IF NO
3348 015736 005716      TST          (SP)                ;:IS THIS THE FIRST RUBOUT?
3349 015740 001007      BNE          6$                  ;:BR IF NO
3350 015742 112737 000134 016140  MOVB      #' \, 9$      ;:TYPE A BACK SLASH
3351 015748 104401 016140      TYPE          9$          ;:
3352 015754 012716 177777      MOV      1-1, (SP)        ;:SET THE RUBOUT KEY
3353 015760 005303 6$:      DEC          R3              ;:BACKUP BY ONE
3354 015762 020327 016142      CMP      R3, #TTYIN        ;:STACK EMPTY?
3355 015766 103434      BLO          4$                  ;:BR IF YES
3356 015770 111337 016140      MOVB      (R3), 9$        ;:SETUP TO TYPEOUT THE DELETED CHAR.
3357 015774 104401 016140      TYPE          9$          ;:GO TYPE
3358 016000 000746      BR          2$                  ;:GO READ ANOTHER CHAR.
3359 016002 005716 5$:      TST          (SP)            ;:RUBOUT KEY SET?
3360 016004 001406      BEQ          7$                  ;:BR IF NO
3361 016006 112737 000134 016140  MOVB      #' \, 9$      ;:TYPE A BACK SLASH
3362 016014 104401 016140      TYPE          9$          ;:
3363 016020 005016      CLR          (SP)                ;:CLEAR THE RUBOUT KEY
3364 016022 122713 000025 7$:      CMPB      #25, (R3)    ;:IS CHARACTER A CTRL U?
3365 016026 001003      BNE          8$                  ;:BR IF NO
3366 016030 104401 016151      TYPE          $CNTLU      ;:TYPE A CONTROL "U"
3367 016034 000726      BR          1$                  ;:GO START OVER
3368 016036 122713 000022 8$:      CMPB      #22, (R3)    ;:IS CHARACTER A "r"?
3369 016042 001011      BNE          3$                  ;:BRANCH IF NO
3370 016044 105013      CLRB          (R3)              ;:CLEAR THE CHARACTER
3371 016046 104401 001207      TYPE          , $CRLF      ;:TYPE A "CR" & "LF"
3372 016052 104401 016142      TYPE          , TTYIN      ;:TYPE THE INPUT STRING
3373 016056 000717      BR          2$                  ;:GO PICKUP ANOTHER CHARACTER
3374 016060 104401 001206 4$:      TYPE          $QUES    ;:TYPE A '?'
3375 016064 000712      BR          1$                  ;:CLEAR THE BUFFER AND LOOP
3376 016066 111337 016140 3$:      MOVB      (R3), 9$      ;:ECHO THE CHARACTER
3377 016072 104401 016140      TYPE          9$          ;:
3378 016076 122723 000015      CMPB      15, (R3)+      ;:CHECK FOR RETURN
3379 016102 001305      BNE          2$                  ;:LOOP IF NOT RETURN
3380 016104 105063 177777      CLRB          -1(R3)       ;:CLEAR RETURN (THE 15)
3381 016110 104401 001210      TYPE          , $LF        ;:TYPE A LINE FEED
3382 016114 005726      TST          (SP)+              ;:CLEAN RUBOUT KEY FROM THE STACK
3383 016116 012603      MOV          (SP)+, R3          ;:RESTORE R3
3384 016120 011646      MOV          (SP), -(SP)        ;:ADJUST THE STACK AND PUT ADDRESS OF THE
3385 016122 016666 000004 000002  MOV      4(SP), 2(SP) ;:FIRST ASCII CHARACTER ON IT

```


Address	Hex	Dec	Label	Instruction	Comment
3386	016130	012766		MOV #STTYIN,4(SP)	
3387	016136	000002		RTI	;; RETURN
3388	016140	000	9\$:	.BYTE 0	;; STORAGE FOR ASCII CHAR. TO TYPE
3389	016141	000		.BYTE 0	;; TERMINATOR
3390	016142	000007	STTYIN:	.BLKB 7	;; RESERVE 7 BYTES FOR TTY INPUT
3391	016151	136	SCNTLU:	.ASCIZ /U/(15)<(12)	;; CONTROL "U"
3392	016156	043536	SCNTLG:	.ASCIZ /G/(15)<(12)	;; CONTROL "G"
3393	016163	015	SMSWR:	.ASCIZ <15><12>/SWR = /	
3394	016170	036440			
3395	016174	020040	SMNEW:	.ASCIZ / NEW = /	
3396	016202	020075			
3397		016206		.EVEN	
3398			.SBTTL	READ AN OCTAL NUMBER FROM THE TTY	
3399					
3400				*****	
3401				*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND	
3402				*CHANGE IT TO BINARY.	
3403				*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL	
3404				*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED	
3405				*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST	
3406				*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.	
3407				*CALL:	
3408				* RDOCT	;; READ AN OCTAL NUMBER
3409				* RETURN HERE	;; LOW ORDER BITS ARE ON TOP OF THE STACK
3410				* *	;; HIGH ORDER BITS ARE IN \$HIOCT
3411					
3412	016206	011646	SRDOCT:	MOV (SP),-(SP)	;; PROVIDE SPACE FOR THE
3413	016210	016666		MOV 4(SP),2(SP)	;; INPUT NUMBER
3414	016216	010046		MOV RO,-(SP)	;; PUSH RO ON STACK
3415	016220	010146		MOV R1,-(SP)	;; PUSH R1 ON STACK
3416	016222	010246		MOV R2,-(SP)	;; PUSH R2 ON STACK
3417	016224	104411	1\$:	RDLIN	;; READ AN ASCII LINE
3418	016226	012600		MOV (SP)+,RO	;; GET ADDRESS OF 1ST CHARACTER
3419	016230	010037		MOV RO,\$\$	;; AND SAVE IT
3420	016234	005001		CLR R1	;; CLEAR DATA WORD
3421	016236	005002		CLR R2	
3422	016240	112046	2\$:	MOVB (RO)+,-(SP)	;; PICKUP THIS CHARACTER
3423	016242	001420		BEQ 3\$	;; IF ZERO GET OUT
3424	016244	122716		CMQB #'0,(SP)	;; MAKE SURE THIS CHARACTER
3425	016250	003026		BGT 4\$	;; IS AN OCTAL DIGIT
3426	016252	122716		CMQB #'7,(SP)	
3427	016256	002423		BLT 4\$	
3428	016260	006301		ASL R1	;; *2
3429	016262	006102		ROL R2	
3430	016264	006301		ASL R1	;; *4
3431	016266	006102		ROL R2	
3432	016270	006301		ASL R1	;; *8
3433	016272	006102		ROL R2	
3434	016274	042716		BIC #'C7,(SP)	;; STRIP THE ASCII JUNK
3435	016300	062601		ADD (SP)+,R1	;; ADD IN THIS DIGIT
3436	016302	000756		BR 2\$	;; LOOP
3437	016304	005726	3\$:	TST (SP)+	;; CLEAN TERMINATOR FROM STACK
3438	016306	010166		MOV R1,12(SP)	;; SAVE THE RESULT
3439	016312	010237		MOV R2,\$HIOCT	
3440	016316	012602		MOV (SP)+,R2	;; POP STACK INTO R2
3441	016320	012601		MOV (SP)+,R1	;; POP STACK INTO R1

```

3442 016322 012600
3443 016324 000002
3444 016326 005726
3445 016330 105010
3446 016332 104401
3447 016334 000000
3448 016336 104401 001206
3449 016342 000730
3450 016344 000000
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468 016346
3469 016346 010046
3470 016350 010146
3471 016352 010246
3472 016354 010346
3473 016356 010446
3474 016360 010546
3475 016362 016646 000022
3476 016366 016646 000022
3477 016372 016646 000022
3478 016376 016646 000022
3479 016402 000002
3480
3481
3482
3483
3484 016404
3485 016404 012666 000022
3486 016410 012666 000022
3487 016414 012666 000022
3488 016420 012666 000022
3489 016424 012605
3490 016426 012604
3491 016430 012603
3492 016432 012602
3493 016434 012601
3494 016436 012600
3495 016440 000002
3496
3497

```

```

MOV (SP)+,R0      ;; POP STACK INTO R0
RTI                ;; RETURN
4S: TST (SP)+      ;; CLEAN PARTIAL FROM STACK
    CLRB (R0)      ;; SET A TERMINATOR
    TYPE          ;; TYPE UP THRU THE BAD CHAR.
5S: .WORD 0
    TYPE $QUES     ;; "?" "CR" & "LF"
    BR 1$          ;; TRY AGAIN
SHIOCT: .WORD 0    ;; HIGH ORDER BITS GO HERE
.SBTTL 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---R0

```

```

$SAVREG:
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 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

```

```

;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
.SBTTL TRAP DECODER

```



```

3498
3499
3500
3501
3502
3503
3504 016442 010046
3505 016444 016600 000002
3506 016450 005740
3507 016452 111000
3508 016454 006300
3509 016456 016000 016476
3510 016462 000200
3511
3512
3513
3514
3515 016464 011646
3516 016466 016666 000004 000002
3517 016474 000002
3518
3519
3520
3521
3522
3523
3524
3525
3526 016476 016464
3527 016500 014150
3528 016502 014414
3529 016504 014370
3530 016506 014430
3531 016510 014616
3532
3533 016512 015404
3534
3535 016514 015314
3536 016516 015616
3537 016520 015706
3538 016522 016206
3539 016524 016346
3540 016526 016404
3541
3542
3543
3544
3545
3546
3547
3548 016530 005015 041412 051132
3549 016536 044115 030102 005015
3550 016544 046522 031460 051057
3551 016552 030115 020062 052504
3552 016560 046101 041440 047117
3553 016566 051124 046117 042514

;*****
;THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
;AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
;OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
;GO TO THAT ROUTINE.
STRAP:  MOV  RO, -(SP)          ;;SAVE 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  STRPAD(RO), RO    ;;INDEX TO TABLE
        RTS  RO               ;;GO TO ROUTINE

;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
STRAP2: MOV  (SP), -(SP)      ;;MOVE THE PC DOWN
        MOV  4(SP), 2(SP)    ;;MOVE THE PSW DOWN
        RTI                  ;;RESTORE THE PSW

.SBTTL  TRAP TABLE
;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
;BY THE "TRAP" INSTRUCTION.
;
ROUTINE
-----
STRPAD:  WORD  STRAP2          TRAP+1(104401)  TTY TYPEOUT ROUTINE
        TYPE  ;;CALL=TYPE
        $TYPOC ;;CALL=TYPOC   TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOS ;;CALL=TYPOS   TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPON ;;CALL=TYPON   TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
        $TYPDS ;;CALL=TYPDS   TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
        SGTSWR ;;CALL=GTSWR   TRAP+6(104406)  GET SOFT-SWR SETTING
        SCKSWR ;;CALL=CKSWR   TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
        $RDCHR ;;CALL=RDCHR   TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
        $RDLIN ;;CALL=RDLIN   TRAP+11(104411) TTY TYPEIN STRING ROUTINE
        $RDOCT ;;CALL=RDOCT   TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
        $SAVREG ;;CALL=SAVREG TRAP+13(104413) SAVE RO-R5 ROUTINE
        $RESREG ;;CALL=RESREG TRAP+14(104414) RESTORE RO-R5 ROUTINE

;*****
.SBTTL  TELETYPE MESSAGES
;*****
TITLE:  .ASCII <CR><LF><LF>/CZRMH80/<CR><LF>
        .ASCII 3RMD3/RMD2 DUAL CONTROLLER LOGIC TEST - PART 22<CR><LF><LF>

```

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 66
TELETYPE MESSAGES

SEQ 0066

3554	016574	020122	047514	044507
3555	016602	020103	042524	052123
3556	016610	026440	050040	051101
3557	016616	020124	006462	005012
3558	016624	000		
3559	016625	015	042412	052116
3560	016632	051105	042040	044522
3561	016640	042526	040440	042104
3562	016646	042522	051523	020072
3563	016654	000		
3564	016655	111	053116	046101
3565	016662	042111	040440	042104
3566	016670	042522	051523	005015
3567	016676	000		
3568	016677	015	050012	051117
3569	016704	020124	020101	042101
3570	016712	051104	051505	020123
3571	016720	051511	020072	000
3572	016725	015	050012	051117
3573	016732	020124	020102	042101
3574	016740	051104	051505	020123
3575	016746	051511	020072	000
3576	016753	015	051412	051531
3577	016760	042524	020115	052515
3578	016766	052123	044040	053101
3579	016774	020105	046047	020047
3580	017002	051117	023440	023520
3581	017010	041440	047514	045503
3582	017016	005015	000012	
3583	017022	042412	052116	051105
3584	017030	052040	051505	020124
3585	017036	035043	000040	
3586	017042	047111	040526	044514
3587	017050	020104	042524	052123
3588	017056	047040	046525	042502
3589	017064	006522	000012	
3590	017070	005015	052012	042510
3591	017076	050040	042522	042523
3592	017104	052116	040440	042104
3593	017112	042522	051523	047440
3594	017120	020106	044124	020105
3595	017126	044122	030461	024040
3596	017134	046522	051503	024461
3597	017142	044440	035123	000040
3598	017150	042412	052116	051105
3599	017156	047040	053505	051040
3600	017164	030510	020061	042101
3601	017172	051104	051505	035123
3602	017200	000040		
3603	017202	005015	050012	042522
3604	017210	051523	023440	052123
3605	017216	047101	041104	023531
3606	017224	047440	020116	051104
3607	017232	053111	105	
3608	017235	015	050012	047522
3609	017242	051107	046501	053440

ENTERA: .ASCIZ <CR><LF>/ENTER DRIVE ADDRESS: /

ADRERR: .ASCIZ /INVALID ADDRESS/<CR><LF>

PORTAIS: .ASCIZ <CR><LF>/PORT A ADDRESS IS: /

PORTBIS: .ASCIZ <CR><LF>/PORT B ADDRESS IS: /

NOCLOCK: .ASCIZ <CR><LF>/SYSTEM MUST HAVE 'L' OR 'P' CLOCK/<CR><LF><LF>

TESTNO: .ASCIZ <LF>/ENTER TEST #: /

BADNO: .ASCIZ /INVALID TEST NUMBER/<CR><LF>

ADDRIS: .ASCIZ <CR><LF><LF>/THE PRESENT ADDRESS OF THE RH11 (RMCS1) IS: /

NTRH11: .ASCIZ <LF>/ENTER NEW RH11 ADDRESS: /

STANDBY: .ASCII <CR><LF><LF>/PRESS 'STANDBY' ON DRIVE/

.ASCIZ <CR><LF>/PROGRAM WILL LOOP WAITING FOR 'MOL' TO SET FROM PORT /

C06

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 67
TELETYPE MESSAGES

SEQ 0067

3610	017250	046111	020114	047514
3611	017256	050117	053440	044501
3612	017264	044524	043516	043040
3613	017272	051117	023440	047515
3614	017300	023514	052040	020117
3615	017306	042523	020124	051106
3616	017314	046517	050040	051117
3617	017322	020124	000	
3618	017325	015	005012	042522
3619	017332	052524	047122	023440
3620	017340	047503	052116	047522
3621	017346	046114	051105	051440
3622	017354	046105	041505	023524
3623	017362	051440	044527	041524
3624	017370	020110	047117	042040
3625	017376	044522	042526	052040
3626	017404	020117	040447	041057
3627	017412	000047		
3628	017414	005015	052012	051125
3629	017422	020116	041447	047117
3630	017430	051124	046117	042514
3631	017436	020122	042523	042514
3632	017444	052103	020047	053523
3633	017452	052111	044103	047440
3634	017460	020116	051104	053111
3635	017466	020105	047524	023440
3636	017474	023501	000	
3637	017477	015	005012	052524
3638	017504	047122	023440	047503
3639	017512	052116	047522	046114
3640	017520	051105	051440	046105
3641	017526	041505	023524	051440
3642	017534	044527	041524	020110
3643	017542	047117	042040	044522
3644	017550	042526	052040	020117
3645	017556	041047	000047	
3646	017562	005015	044124	047105
3647	017570	050040	042522	051523
3648	017576	023440	047503	052116
3649	017604	020047	047117	052040
3650	017612	042510	050040	047522
3651	017620	042503	051523	051117
3652	017626	005015	000	
3653	017631	015	005012	052123
3654	017636	050117	052040	042510
3655	017644	042040	044522	042526
3656	017652	000		
3657	017653	015	005012	052123
3658	017660	051101	020124	044124
3659	017666	020105	051104	053111
3660	017674	020105	020055	044124
3661	017702	020105	051120	043517
3662	017710	040522	020115	044527
3663	017716	046114	053440	044501
3664	017724	020124	047506	020122
3665	017732	046447	046117	020047

SWTCHN: .ASCIZ <CR><LF><LF>RETURN 'CONTROLLER SELECT' SWITCH ON DRIVE TO 'A/B'>

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

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/>

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34

MACY11 30(1046) 21-NOV-77 14:15 PAGE 68
TELETYPE MESSAGES

SEQ 0068

3666 017740 047524 051440 052105
3667 017746 000

3668
3669
3670
3671
3672
3673
3674

;;*****

.SBTTL TEST ERROR MESSAGES

;;*****

3675 017747 104 044522 042526
3676 017754 044440 020123 047516
3677 017762 026516 054105 051511
3678 017770 042524 052116 024040
3679 017776 047047 042105 020047
3680 020004 044502 020124 042523
3681 020012 024524 000

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

3682
3683 020015 127 047522 043516
3684 020022 042040 044522 042526
3685 020030 052040 050131 000105

EM2: .ASCIZ /WRONG DRIVE TYPE/

3686
3687 020036 047503 052116 047522
3688 020044 046114 051105 051440
3689 020052 046105 041505 020124
3690 020060 053523 052111 044103
3691 020066 047440 020116 051104
3692 020074 053111 020105 047516
3693 020102 020124 047111 023440
3694 020110 027501 023502 000

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

3695
3696 020115 104 044522 042526
3697 020122 047040 052117 047440
3698 020130 020116 044514 042516
3699 020136 000

EM4: .ASCIZ /DRIVE NOT ON LINE/

3700
3701 020137 123 051105 040511
3702 020144 020114 052516 041115
3703 020152 051105 051040 040505
3704 020160 020104 044124 047522
3705 020166 043525 020110 040505
3706 020174 044103 050040 051117
3707 020202 020124 047516 020124
3708 020210 044124 020105 040523
3709 020216 042515 000

EM5: .ASCIZ /SERIAL NUMBER READ THROUGH EACH PORT NOT THE SAME/

3710
3711 020221 124 046511 047505
3712 020226 052125 044040 051501
3713 020234 047040 052117 047440
3714 020242 041503 051125 042522
3715 020250 020104 044527 044124
3716 020256 047111 031040 051440
3717 020264 041505 047117 051504
3718 020272 000

EM6: .ASCIZ /TIMEOUT HAS NOT OCCURRED WITHIN 2 SECONDS/

3719
3720 020273 124 046511 047505
3721 020300 052125 047440 042516

EM7: .ASCIZ /TIMEOUT ONE-SHOT IS LESS THAN 500 MS/

E06

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 69
TEST ERROR MESSAGES

SEQ 0069

3722	020306	051455	047510	020124	
3723	020314	051511	046040	051505	
3724	020322	020123	044124	047101	
3725	020330	032440	030060	046440	
3726	020336	000123			
3727					
3728	020340	042522	042101	047111	EM10: .ASCIZ /READIN PRESET DOES NOT SET VOLUME VALID FOR THE PORT/
3729	020346	050040	042522	042523	
3730	020354	020124	047504	051505	
3731	020362	047040	052117	051440	
3732	020370	052105	053040	046117	
3733	020376	046525	020105	040526	
3734	020404	044514	020104	047506	
3735	020412	020122	044124	020105	
3736	020420	047520	052122	000	
3737					
3738	020425	047	047507	020047	EM11: .ASCIZ /'GO' BIT RESET DURING UNLOAD COMMAND/
3739	020432	044502	020124	042522	
3740	020440	042523	020124	052504	
3741	020446	044522	043516	052440	
3742	020454	046116	040517	020104	
3743	020462	047503	046515	047101	
3744	020470	000104			
3745					
3746	020472	047111	047503	051122	EM12: .ASCIZ /INCORRECT STATUS DURING UNLOAD COMMAND/
3747	020500	041505	020124	052123	
3748	020506	052101	051525	042040	
3749	020514	051125	047111	020107	
3750	020522	047125	047514	042101	
3751	020530	041440	046517	040515	
3752	020536	042116	000		
3753					
3754	020541	104	044522	042526	EM13: .ASCIZ /DRIVE DID NOT RETURN TO NEUTRAL AFTER UNLOAD COMMAND/
3755	020546	042040	042111	047040	
3756	020554	052117	051040	052105	
3757	020562	051125	020116	047524	
3758	020570	047040	052505	051124	
3759	020576	046101	040440	052106	
3760	020604	051105	052440	046116	
3761	020612	040517	020104	047503	
3762	020620	046515	047101	000104	
3763					
3764	020626	052101	042524	052116	EM14: .ASCIZ /ATTENTION BIT SET ON 'OPPOSITE PORT' AFTER UNLOAD/
3765	020634	047511	020116	044502	
3766	020642	020124	042523	020124	
3767	020650	047117	023440	050117	
3768	020656	047520	044523	042524	
3769	020664	050040	051117	023524	
3770	020672	040440	052106	051105	
3771	020700	052440	046116	040517	
3772	020706	000104			
3773					
3774	020710	052101	042524	052116	EM15: .ASCIZ /ATTENTION BIT NOT SET ON PORT WHICH ISSUED 'UNLOAD'/
3775	020716	047511	020116	044502	
3776	020724	020124	047516	020124	
3777	020732	042523	020124	047117	

F06

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 70
TEST ERROR MESSAGES

SEQ 0070

3778	020740	050040	051117	020124	
3779	020746	044127	041511	020110	
3780	020754	051511	052523	042105	
3781	020762	023440	047125	047514	
3782	020770	042101	000047		
3783					
3784	020774	051104	053111	020105	EM16: .ASCII /DRIVE NOT IN NEUTRAL AFTER UNLOAD WITH 'CONTROLLER'<CR><LF>
3785	021002	047516	020124	047111	
3786	021010	047040	052505	051124	
3787	021016	046101	040440	052106	
3788	021024	051105	052440	046116	
3789	021032	040517	020104	044527	
3790	021040	044124	023440	047503	
3791	021046	052116	047522	046114	
3792	021054	051105	005015		
3793	021060	042523	042514	052103	.ASCIIZ 'SELECT' SWITCH MOVED FROM 'A/B'>
3794	021066	020047	053523	052111	
3795	021074	044103	046440	053117	
3796	021102	042105	043040	047522	
3797	021110	020115	040447	041057	
3798	021116	000047			
3799					
3800	021120	051104	053111	020105	EM17: .ASCIIZ /DRIVE LOCKED ON PORT 'A' BY SWITCH WHILE CYCLED UP/
3801	021126	047514	045503	042105	
3802	021134	047440	020116	047520	
3803	021142	052122	023440	023501	
3804	021150	041040	020131	053523	
3805	021156	052111	044103	053440	
3806	021164	044510	042514	041440	
3807	021172	041531	042514	020104	
3808	021200	050125	000		
3809					
3810	021203	104	044522	042526	EM20: .ASCIIZ /DRIVE LOCKED ON PORT 'B' BY SWITCH WHILE CYCLED UP/
3811	021210	046040	041517	042513	
3812	021216	020104	047117	050040	
3813	021224	051117	020124	041047	
3814	021232	020047	054502	051440	
3815	021240	044527	041524	020110	
3816	021246	044127	046111	020105	
3817	021254	054503	046103	042105	
3818	021262	052440	000120		
3819					
3820	021266	052123	052101	051525	EM21: .ASCIIZ /STATUS INCORRECT FOR PORT AFTER CYCLE UP/
3821	021274	044440	041516	051117	
3822	021302	042522	052103	043040	
3823	021310	051117	050040	051117	
3824	021316	020124	043101	042524	
3825	021324	020122	054503	046103	
3826	021332	020105	050125	000	
3827					
3828	021337	122	043505	051511	EM22: .ASCIIZ /REGISTER CONTENTS SEEN WHEN DRIVE SWITCHED ON 'OPPOSITE' PORT/
3829	021344	042524	020122	047503	
3830	021352	052116	047105	051524	
3831	021360	051440	042505	020116	
3832	021366	044127	047105	042040	
3833	021374	044522	042526	051440	

G06

CZRMHBO RM03/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 71
TEST ERROR MESSAGES

SEQ 0071

3834	021402	044527	041524	042510	
3835	021410	020104	047117	023440	
3836	021416	050117	047520	044523	
3837	021424	042524	020047	047520	
3838	021432	052122	000		
3839					
3840	021435	047	042516	023504	EM23: .ASCIZ /'NED' NOT SET WHEN RMDS1 ACCESSED THROUGH PORT NOT SWITCHED/
3841	021442	047040	052117	051440	
3842	021450	052105	053440	042510	
3843	021456	020116	046522	051504	
3844	021464	020061	041501	042503	
3845	021472	051523	042105	052040	
3846	021500	051110	052517	044107	
3847	021506	050040	051117	020124	
3848	021514	047516	020124	053523	
3849	021522	052111	044103	042105	
3850	021530	000			
3851					
3852	021531	104	044522	042526	EM24: .ASCIZ /DRIVE SWITCHED TO LOCKED OUT PORT WHEN RELEASED/
3853	021536	051440	044527	041524	
3854	021544	042510	020104	047524	
3855	021552	046040	041517	042513	
3856	021560	020104	052517	020124	
3857	021566	047520	052122	053440	
3858	021574	042510	020116	042522	
3859	021602	042514	051501	042105	
3860	021610	000			
3861					
3862	021611	122	030510	020061	EM25: .ASCIZ /RH11 DIDN'T RESPOND TO ADDRESSING/
3863	021616	044504	047104	052047	
3864	021624	051040	051505	047520	
3865	021632	042116	052040	020117	
3866	021640	042101	051104	051505	
3867	021646	044523	043516	000	
3868					
3869	021653	104	044522	042526	EM30: .ASCIZ /DRIVE NOT SEIZED BY PORT/
3870	021660	047040	052117	051440	
3871	021666	044505	042532	020104	
3872	021674	054502	050040	051117	
3873	021702	000124			
3874					
3875	021704	051127	047117	020107	EM31: .ASCIZ /WRONG STATUS SEEN BY THE SEIZING PORT/
3876	021712	052123	052101	051525	
3877	021720	051440	042505	020116	
3878	021726	054502	052040	042510	
3879	021734	051440	044505	044532	
3880	021742	043516	050040	051117	
3881	021750	000124			
3882					
3883	021752	042522	044507	052123	EM32: .ASCIZ /REGISTER CONTENTS WRONG/
3884	021760	051105	041440	047117	
3885	021766	042524	052116	020123	
3886	021774	051127	047117	000107	
3887					
3888	022002	047503	052116	047522	EM33: .ASCIZ /CONTROL BUS PARITY ERROR READING INDICATED REGISTER/
3889	022010	020114	052502	020123	

H06

CZRMH80 RM03/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 72
TEST ERROR MESSAGES

SEQ 0072

3890	022016	040520	044522	054524	
3891	022024	042440	051122	051117	
3892	022032	051040	040505	044504	
3893	022040	043516	044440	042116	
3894	022046	041511	052101	042105	
3895	022054	051040	043505	051511	
3896	022062	042524	000122		
3897					
3898	022066	040503	023516	020124	EM34: .ASCIZ /CAN'T ACCESS DRIVE THROUGH EITHER PORT/
3899	022074	041501	042503	051523	
3900	022102	042040	044522	042526	
3901	022110	052040	051110	052517	
3902	022116	044107	042440	052111	
3903	022124	042510	020122	047520	
3904	022132	052122	000		
3905					
3906	022135	104	044522	042526	EM35: .ASCIZ /DRIVE NOT IN NEUTRAL AFTER RELEASE - REQUEST NOT SET/
3907	022142	047040	052117	044440	
3908	022150	020116	042516	052125	
3909	022156	040522	020114	043101	
3910	022164	042524	020122	042522	
3911	022172	042514	051501	020105	
3912	022200	020055	042522	052521	
3913	022206	051505	020124	047516	
3914	022214	020124	042523	000124	
3915					
3916	022222	051104	053111	020105	EM36: .ASCIZ /DRIVE NOT IN NEUTRAL AFTER TIMEOUT - REQUEST NOT SET/
3917	022230	047516	020124	047111	
3918	022236	047040	052505	051124	
3919	022244	046101	040440	052106	
3920	022252	051105	052040	046511	
3921	022260	047505	052125	026440	
3922	022266	051040	050505	042525	
3923	022274	052123	047040	052117	
3924	022302	051440	052105	000	
3925					
3926	022307	122	043505	051511	EM37: .ASCIZ /REGISTER CONTENTS WRONG AFTER RELEASE OR TIMEOUT/
3927	022314	042524	020122	047503	
3928	022322	052116	047105	051524	
3929	022330	053440	047522	043516	
3930	022336	040440	052106	051105	
3931	022344	051040	046105	040505	
3932	022352	042523	047440	020122	
3933	022360	044524	042515	052517	
3934	022366	000124			
3935					
3936	022370	051104	053111	020105	EM40: .ASCIZ /DRIVE IN NEUTRAL AFTER RELEASE - REQUEST SET/
3937	022376	047111	047040	052505	
3938	022404	051124	046101	040440	
3939	022412	052106	051105	051040	
3940	022420	046105	040505	042523	
3941	022426	026440	051040	050505	
3942	022434	042525	052123	051440	
3943	022442	052105	000		
3944					
3945	022445	122	043505	051511	EM41: .ASCIZ /REGISTER WRONG AFTER RELEASE WITH REQUEST SET/

CZRMH80 RMO3/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34

MACY11 30(1046) 21-NOV-77 14:15 PAGE 73
TEST ERROR MESSAGES

SEQ 0073

3946	022452	042524	020122	051127					
3947	022460	047117	020107	043101					
3948	022466	042524	020122	042522					
3949	022474	042514	051501	020105					
3950	022502	044527	044124	051040					
3951	022510	050505	042525	052123					
3952	022516	051440	052105	000					
3953									
3954									
3955	022523	124	051505	020124	DH1:	.ASCIZ	/TEST #	ERR PC	PORT # REG ADR CONTENTS/
3956	022530	020043	042440	051122					
3957	022536	050040	020103	050040					
3958	022544	051117	020124	020043					
3959	022552	051040	043505	040440					
3960	022560	051104	041440	047117					
3961	022566	042524	052116	000123					
3962	022574	042524	052123	021440	DH2:	.ASCIZ	/TEST #	ERR PC	PORT # REG ADR GOOD BAD/
3963	022602	020040	051105	020122					
3964	022610	041520	020040	047520					
3965	022616	052122	021440	020040					
3966	022624	042522	020107	042101					
3967	022632	020122	047507	042117					
3968	022640	020040	020040	040502					
3969	022646	000104							
3970	022650	042524	052123	021440	DH5:	.ASCIZ	/TEST #	ERR PC	REG ADR PORT A PORT B/
3971	022656	020040	051105	020122					
3972	022664	041520	020040	042522					
3973	022672	020107	042101	020122					
3974	022700	047520	052122	040440					
3975	022706	020040	047520	052122					
3976	022714	041040	000						
3977	022717	124	051505	020124	DH6:	.ASCIZ	/TEST #	ERR PC	PORT #/
3978	022724	020043	042440	051122					
3979	022732	050040	020103	050040					
3980	022740	051117	020124	000043					
3981	022746	042524	052123	021440	DH7:	.ASCIZ	/TEST #	ERR PC	PORT # TIME (IN MS)/
3982	022754	020040	051105	020122					
3983	022762	041520	020040	047520					
3984	022770	052122	021440	020040					
3985	022776	044524	042515	024040					
3986	023004	047111	046440	024523					
3987	023012	000							
3988	023013	040	020040	020040	DH13:	.ASCII	/		SEIZE/<CR><LF>
3989	023020	020040	020040	020040					
3990	023026	020040	020040	051440					
3991	023034	044505	042532	005015					
3992	023042	042524	052123	021440		.ASCIZ	/TEST #	ERR PC	PORT #/
3993	023050	020040	051105	020122					
3994	023056	041520	020040	047520					
3995	023064	052122	021440	000					
3996	023071	040	020040	020040	DH14:	.ASCII	/		SEIZE ERROR/<CR><LF>
3997	023076	020040	020040	020040					
3998	023104	020040	020040	051440					
3999	023112	044505	042532	020040					
4000	023120	042440	051122	051117					
4001	023126	005015							

	023130	042524	052123	021440	.ASCIZ /TEST # ERR PC PORT # PORT # REG ADR CONTENTS/
4002	023130	042524	052123	021440	
4003	023136	020040	051105	020122	
4004	023144	041520	020040	047520	
4005	023152	052122	021440	020040	
4006	023160	047520	052122	021440	
4007	023166	020040	042522	020107	
4008	023174	042101	020122	047503	
4009	023202	052116	047105	051524	
4010	023210	000			
4011	023211	124	051505	020124	DH17: .ASCIZ /TEST # ERR PC/
4012	023216	020043	042440	051122	
4013	023224	050040	000103		
4014	023230	020040	020040	020040	DH24: .ASCII / LOCKED SWITCHED TO/<CR><LF>
4015	023236	020040	020040	020040	
4016	023244	020040	020040	047514	
4017	023252	045503	042105	020040	
4018	023260	053523	052111	044103	
4019	023266	042105	052040	006517	
4020	023274	012			
4021	023275	124	051505	020124	.ASCIZ /TEST # ERR PC PORT # PORT #/
4022	023302	020043	042440	051122	
4023	023310	050040	020103	050040	
4024	023316	051117	020124	020043	
4025	023324	050040	051117	020124	
4026	023332	000043			
4027	023334	051044	040515	051104	DH25: .ASCIZ /\$RMADR/
4028	023342	000			
4029	023343	040	020040	020040	DH30: .ASCII / SEIZE ERROR/<CR><LF>
4030	023350	020040	020040	020040	
4031	023356	020040	020040	051440	
4032	023364	044505	042532	020040	
4033	023372	042440	051122	051117	
4034	023400	005015			
4035	023402	042524	052123	021440	.ASCIZ /TEST # ERR PC PORT # PORT # REG ADR GOOD BAD/
4036	023410	020040	051105	020122	
4037	023416	041520	020040	047520	
4038	023424	052122	021440	020040	
4039	023432	047520	052122	021440	
4040	023440	020040	042522	020107	
4041	023446	042101	020122	047507	
4042	023454	042117	020040	020040	
4043	023462	040502	000104		
4044	023466	020040	020040	020040	DH34: .ASCII / PORT A PORT B/<CR><LF>
4045	023474	020040	020040	020040	
4046	023502	020040	020040	047520	
4047	023510	052122	040440	020040	
4048	023516	047520	052122	041040	
4049	023524	005015			
4050	023526	042524	052123	021440	.ASCIZ /TEST # ERR PC RMDS1 RMDS1/
4051	023534	020040	051105	020122	
4052	023542	041520	020040	046522	
4053	023550	051504	020061	020040	
4054	023556	046522	051504	000061	
4055	023564	020040	020040	020040	DH35: .ASCII / RELSNG ERROR/<CR><LF>
4056	023572	020040	020040	020040	
4057	023600	020040	020040	042522	

K06

CZRMHBO RM03/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 75
TEST ERROR MESSAGES

SEQ 0075

4058	023606	051514	043516	020040					
4059	023614	051105	047522	006522					
4060	023622	012							
4061	023623	124	051505	020124		.ASCIZ	/TEST #	ERR PC	PORT # PORT #/
4062	023630	020043	042440	051122					
4063	023636	050040	020103	050040					
4064	023644	051117	020124	020043					
4065	023652	050040	051117	020124					
4066	023660	000043							
4067	023662	020040	020040	020040	DH37:	.ASCII	/	RELSNG	ERROR/<CR><LF>
4068	023670	020040	020040	020040					
4069	023676	020040	020040	042522					
4070	023704	051514	043516	020040					
4071	023712	051105	047522	006522					
4072	023720	012							
4073	023721	124	051505	020124		.ASCIZ	/TEST #	ERR PC	PORT # PORT # REG ADR GOOD BAD/
4074	023726	020043	042440	051122					
4075	023734	050040	020103	050040					
4076	023742	051117	020124	020043					
4077	023750	050040	051117	020124					
4078	023756	020043	051040	043505					
4079	023764	040440	051104	043440					
4080	023772	047517	020104	020040					
4081	024000	041040	042101	000					
4082	024005	040	020040	020040	DH40:	.ASCII	/	RELSNG	RQSTNG/<CR><LF>
4083	024012	020040	020040	020040					
4084	024020	020040	020040	051040					
4085	024026	046105	047123	020107					
4086	024034	051040	051521	047124					
4087	024042	006507	012						
4088	024045	124	051505	020124		.ASCIZ	/TEST #	ERR PC	PORT # PORT #/
4089	024052	020043	042440	051122					
4090	024060	050040	020103	050040					
4091	024066	051117	020124	020043					
4092	024074	050040	051117	020124					
4093	024102	000043							
4094									
4095									
4096						.EVEN			
4097	024104	001242	001116	001234	DT1:	.WORD	TSTNUM,	SERRPC,	PTNBR,\$BDADR,\$BDDAT,0
4098	024112	001122	001126	000000					
4099	024120	001242	001116	001234	DT2:	.WORD	TSTNUM,	SERRPC,	PTNBR,\$BDADR,\$GDDAT,\$BDDAT,0
4100	024126	001122	001124	001126					
4101	024134	000000							
4102	024136	001242	001116	001122	DT5:	.WORD	TSTNUM,	SERRPC,\$BDADR,\$GDDAT,\$BDDAT,0	
4103	024144	001124	001126	000000					
4104	024152	001242	001116	001234	DT6:	.WORD	TSTNUM,	SERRPC,PTNBR,0	
4105	024160	000000							
4106	024162	001242	001116	001234	DT7:	.WORD	TSTNUM,	SERRPC,PTNBR,TIME,0	
4107	024170	001252	000000						
4108	024174	001242	001116	001236	DT13:	.WORD	TSTNUM,	SERRPC,SEIZPT,0	
4109	024202	000000							
4110	024204	001242	001116	001236	DT14:	.WORD	TSTNUM,	SERRPC,SEIZPT,PTNBR,\$BDADR,\$BDDAT,0	
4111	024212	001234	001122	001126					
4112	024220	000000							
4113	024222	001242	001116	000000	DT17:	.WORD	TSTNUM,	SERRPC,0	

4114	024230	001242	001116	001236	DT24:	.WORD	TSTNUM,SERRPC,SEIZPT,PTNBR,0
4115	024236	001234	000000				
4116	024242	001272	000000		DT25:	.WORD	\$RMADR,0
4117	024246	001242	001116	001236	DT30:	.WORD	TSTNUM,SERRPC,SEIZPT,PTNBR,\$BDADR,\$GDDAT,\$BDDAT,0
4118	024254	001234	001122	001124			
4119	024262	001126	000000				
4120	024266	001242	001116	001170	DT34:	.WORD	TSTNUM,SERRPC,STMP2,STMP3,0
4121	024274	001172	000000				
4122	024300	001242	001116	001236	DT35:	.WORD	TSTNUM,SERRPC,SEIZPT,PTNBR,0
4123	024306	001234	000000				
4124	024312	001242	001116	001236	DT40:	.WORD	TSTNUM,SERRPC,SEIZPT,OPPR,0
4125	024320	001240	000000				
4126							
4127	024324	000	000	001	DF1:	.BYTE	0,0,1,0,0
4128	024327	000	000				
4129	024331	000	000	001	DF2:	.BYTE	0,0,1,0,0,0
4130	024334	000	000	000			
4131	024337	000	000	000	DF5:	.BYTE	0,0,0,0,0
4132	024342	000	000				
4133	024344	000	000	001	DF6:	.BYTE	0,0,1
4134	024347	000	000	001	DF7:	.BYTE	0,0,1,1
4135	024352	001					
4136	024353	000	000	001	DF14:	.BYTE	0,0,1,1,0,0
4137	024356	001	000	000			
4138	024361	000	000		DF17:	.BYTE	0,0
4139	024363	000			DF25:	.BYTE	0
4140	024364	000	000	001	DF30:	.BYTE	0,0,1,1,0,0,0
4141	024367	001	000	000			
4142	024372	000					
4143	024373	000	000	000	DF34:	.BYTE	0,0,0,0
4144	024376	000					
4145							
4146	024400				.EVEN		
4147							
4148							
4149							
4150							
4151							
4152							
4153							
4154							
4155							
4156							
4157	024400	002722		TSTADR:	.WORD	TST1+2	: STARTING ADDRESS OF TEST 1
4158	024402	004342			.WORD	TST2+2	: STARTING ADDRESS OF TEST 2
4159	024404	005124			.WORD	TST3+2	: STARTING ADDRESS OF TEST 3
4160	024406	005706			.WORD	TST4+2	: STARTING ADDRESS OF TEST 4
4161	024410	006624			.WORD	TST5+2	: STARTING ADDRESS OF TEST 5
4162	024412	007542			.WORD	TST6+2	: STARTING ADDRESS OF TEST 6
4163	024414	010610			.WORD	TST7+2	: STARTING ADDRESS OF TEST 7
4164	024416	011724			.WORD	TST10+2	: STARTING ADDRESS OF TEST 10
4165							
4166							
4167							
4168	024420	001					
4169	024421	002					

;*****
 .SBTTL CONSTANTS, TABLES, ETC
 ;*****
 ;TABLE OF TEST STARTING ADDRESSES
 ;ATTENTION BIT TABLE
 ATABIT: .BYTE 1 ;ATTENTION BIT FOR DRIVE 0
 .BYTE 2 ;ATTENTION BIT FOR DRIVE 1

M06

CZRMHBO RM03/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34

MACY11 30(1046) 21-NOV-77 14:15 PAGE 77
CONSTANTS, TABLES, ETC

SEQ 0077

4170 024422 004
4171 024423 010
4172 024424 020
4173 024425 040
4174 024426 100
4175 024427 200
4176
4177 024430 000010
4178
4179 000001

.BYTE 4
.BYTE 10
.BYTE 20
.BYTE 40
.BYTE 100
.BYTE 200
MAXTN: .WORD \$TN-1
.END

;ATTENTION BIT FOR DRIVE 2
;ATTENTION BIT FOR DRIVE 3
;ATTENTION BIT FOR DRIVE 4
;ATTENTION BIT FOR DRIVE 5
;ATTENTION BIT FOR DRIVE 6
;ATTENTION BIT FOR DRIVE 7
;MAXIMUM TEST NUMBER

1584*
2402*
2651*

2377 2371 1319
622#
814#
839#
840#
841#

CZRMH80 RMO3/2 DU POR LGC 2
CZRMH8.P11 21-NOV-77 13:34

MACY11 30(1046) 21-NOV-77 14:15 PAGE 81
CROSS REFERENCE TABLE -- USER SYMBOLS

DT03	=	000010	8428
DT04	=	000020	8438
DT05	=	000040	8448
DT06	=	000100	8458
DT07	=	000200	8468
DT08	=	000400	8478
DT1		024104	1055
DT13		024174	1125
DT14		024204	1132
DT17		024222	1153
DT2		024120	1062
DT24		024230	1188
DT25		024242	1195
DT30		024246	1216
DT34		024266	1244
DT35		024300	1251
DT40		024312	1272
DT5		024136	1083
DT6		024152	1090
DT7		024162	1097
DVA	=	004000	7798
DVC	=	000200	8648
ECH	=	000100	8088
ECI	=	004000	8758
EMTVEC	=	000030	7118
EM1		017747	1053
EM10		020340	1102
EM11		020425	1109
EM12		020472	1116
EM13		020541	1123
EM14		020626	1130
EM15		020710	1137
EM16		020774	1144
EM17		021120	1151
EM2		020015	1060
EM20		021203	1158
EM21		021266	1165
EM22		021337	1172
EM23		021435	1179
EM24		021531	1186
EM25		021611	1193
EM3		020036	1067
EM30		021653	1214
EM31		021704	1221
EM32		021752	1228
EM33		022002	1235
EM34		022066	1242
EM35		022135	1249
EM36		022222	1256
EM37		022307	1263
EM4		020115	1074
EM40		022370	1270
EM41		022445	1277
EM5		020137	1081
EM6		020221	1088
EM7		020273	1095

[illegible]

CZRMHBO RMO3/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34

MACY11 30(1046) 21-NOV-77 14:15 PAGE 83
CROSS REFERENCE TABLE -- USER SYMBOLS

```
PGE      = 002000
PGM      = 001000
```

```
PIP      = 020000
PIRQ     = 177772
PIRQVE   = 000240
PORTA    001224
```

PORTAI	016677
PORTB	001226

```

PORTBI      016725
PORTC       001230
PR0          = 000000
PR1          = 000040
PR2          = 000100
PR3          = 000140
PR4          = 000200
PR5          = 000240
PR6          = 000300
PR7          = 000340
PS           = 177776
PSEL         = 002000
PSW          = 177776
PTNBR       001234

```

```

PWVREC= 000024
RDCHR  = 104410
RDLIN  = 104411
ROOCT  = 104412
RDY    = 000200
RELERR = 001250
RELOK  = 000001
RESREG= 104414
RESVEC= 000010
RMAS   = 000016
RMB    = 000004
RMCS1  = 000000

```

RMCS2 = 000010

754#	1575	1577	1581	1607	1609	1613	1694	1719	1804	1829	1931	1974
792#	2119	2213	2264									
2076												
796#												
621#												
715#												
1005#	1347*	1348	1352	1355	1359	1371	1374	1481	1482	1527	1528	1569
1570	1637	1677	1678	1710	1711	1720	1739	1741	1747	1830	1849	1851
1857	1913	1914	1928	1929	1975	1994	1996	2004	2061	2062	2063	2089
2090	2120	2139	2141	2149	2196	2214	2233	2235	2243	2265	2284	2286
2294	2352	2353	2438	2439	2441	2476	2477	2594	2595	2636	2637	2681
2682												
1358	3568#											
1006#	1352*	1353*	1354*	1357*	1365	1502	1503	1546	1547	1601	1602	1639
1724	1735	1736	1753	1787	1788	1820	1821	1834	1845	1846	1863	1916
1917	1918	1944	1945	1979	1990	1991	2010	2058	2059	2073	2074	2124
2135	2136	2155	2200	2218	2229	2230	2249	2269	2280	2281	2300	2400
2401	2442	2443	2489	2490	2546	2547	2632	2633	2635	2668	2669	
1364	3572#											
1007#	1371*	1372*	1373*									
638#												
639#												
640#												
641#												
642#												
643#												
644#												
645#												
618#	619	1389*										
731#												
619#												
1009#	1482*	1503*	1528*	1547*	1570*	1602*	1678*	1711*	1735*	1739*	1747*	1753*
1788#	1821*	1845*	1849*	1857*	1863*	1917*	1929*	1945*	1990*	1994*	2004*	2010*
2062*	2074*	2090*	2135*	2139*	2149*	2155*	2229*	2233*	2243*	2249*	2280*	2284*
2294*	2300*	2353*	2401*	2439*	2443*	2477*	2490*	2547*	2595*	2633*	2637*	2669*
2682*	4097	4099	4104	4106	4110	4114	4117	4122				
710#												
3344	3536#											
3417	3537#											
1346	1402	1428	3538#									
728#												
1015#	1716*	1744*	1826*	1854*	1971*	1999*	2116*	2144*	2210*	2238*	2261*	2289*
1284#	1745	1855	1871#	2000	2145	2239	2290					
3540#												
705#												
903#												
898#												
896#	1683*	1684*	1712*	1793*	1794*	1822*	2000*	2001*	2145*	2146*	2198*	

[illegible]

SEQ 0084

SEQ 0085

[illegible]

SERTTL	001112		960*	2727	2731*	2870*	2889												
SESCAP	001200		992*	1311*															
SFILLC	001156		981*	2984	3015														
SFILLS	001155		980*	3015															
SGDADR	001120		964*																
SGDDAT	001124		966*																
			1575*	1488*	1491	1495*	1509*	1512	1516*	1533*	1538	1542*	1552*	1557	1561*				
			1641	1578	1582*	1590*	1593	1597*	1607*	1610	1614*	1622*	1625	1629*	1638*				
			1922*	1694*	1697	1701*	1719*	1749	1755	1804*	1807	1811*	1829*	1859	1865				
			2119*	1923	1931*	1932	1936	1974*	2006	2012	2067*	2068	2076*	2077	2081				
			2409	2151	2157	2213*	2245	2251	2264*	2296	2302	2377*	2380	2384*	2406*				
			2607*	2413*	2421*	2424	2428*	2448*	2451	2455*	2571*	2574	2578*	2600*	2603				
			2732*	2615*	2618	2622*	2642*	2645	2649*	4099	4102	4117							
\$GET42	013216		3257*	3533															
\$GTSWR	015404		593																
\$HD =	000000		3439*	3450*															
\$HIOCT	016344		957*	1414*	2837*	2838	2840*	2847											
\$ICNT	001104		971*	3254*	3285	3397													
\$INTAG	001135		961*	2873*	2889	2900													
\$ITEMB	001114		996*	2889	3015	3381	3391	3451											
\$LF	001210		1001*	2760*															
\$LKCSB	001214		1000*	2756	2761*														
\$LKCSR	001212		1003*	2765	2769*														
\$LKS	001220		1004*	2766															
\$LLVEC	001222		958*	1313*	1399*	1473*	1673*	1783*	1896*	2041*	2189*	2348*	2542*	2843*	2845				
\$LPADR	001106		2847																
\$LPERR	001110		959*	1314*	1400*	1474*	1674*	1784*	1897*	2042*	2190*	2349*	2468	2543*	2662				
\$LPVEC	001216		1002*	2757															
\$MAIL =	***** U		1331	1338	2843	2879	2968												
\$MNEW	016174		3260	3395*															
\$MSWR	016163		3257	3393*															
\$MXCNT	013640		2841	2847*															
\$NULL	001154		979*	2986	3015														
\$NWTST=	000001		1453*	1455	1651*	1653	1761*	1763	1874*	1876	2019*	2021	2165*	2167	2				

STTYIN 016142
STYPBN= ***** U

1284#	2842#											
2824#	935#											
930#	593											
582#	1311	599	599	600	601	602	603	604	991	992	993	1310
1311	1313	1314	1475	1675	1785	1898	2043	2191	2350	2544	2699	2707
2734	2740	2742	2801#	2807	2808	2809	2810	2815	2827	2829	2830	2831
2832	2833	2844	2847	2854	2855	2856	2857	2858	2867	2874	2879	2883
2889												
2810												
991#	1310*	1475*	1675*	1785*	1898*	2043*	2191*	2350*	2467*	2544*	2661*	2707*
2832#	2838	2841*	2847									
976#	3162	3183	3192	3211	3239	3266						
3163#	3178*	3200	3217*	3323	3325*							
1334	1395	3178#	3252									
3167#	3225	3328										
3164#	3179*	3180	3223*	3224*	3225	3227*						
3165#	3180*	3326	3327*	3328	3330*							
3166#	3179	3227	3330									
975#	3162	3184*	3207*	3209	3215*	3237	3253*	3263	3287*			
3181	3192#											
986#	1489*	1490*	1491	1510*	1511*	1512	1536*	1537*	1538	1555*	1556*	1557
1576#	1577*	1578	1591*	1592*	1593	1608*	1609*	1610	1623*	1624*	1625	1695*
1696#	1697	1722*	1723*	1728	1730	1737	1805*	1806*	1807	1832*	1833*	1838
1840	1847	1934*	1935*	1936	1977*	1978*	1983	1985	1992	2079*	2080*	2081
2122*	2123*	2128	2130	2137	2216*	2217*	2222	2224	2231	2267*	2268*	2273
2275	2282	2378*	2379*	2380	2407*	2408*	2409	2422*	2423*	2424	2449*	2450*
2451	2572*	2573*	2574	2601*	2602*	2603	2616*	2617*	2618	2643*	2644*	2645
987#	1726*	1727*	1728	1742	1836*	1837*	1838	1852	1932*	1933*	1935	1981*
1982*	1983	1997	2077*	2078*	2080	2126*	2127*	2128	2142	2220*	2221*	2222
2236	2271*	2272*	2273	2287								
988#	1721*	1722	1734	1746	1748*	1749	1831*	1832	1844	1856	1858*	1859
1976*	1977	1989	2003	2005*	2006	2121*	2122	2134	2148	2150*	2151	2215*
2216	2228	2242	2244*	2245	2266*	2267	2279	2293	2295*	2296	4120	
989#	1725*	1726	1740	1752	1754*	1755	1835*	1836	1850	1862	1864*	1865
1980*	1981	1995	2009	2011*	2012	2125*	2126	2140	2154	2156*	2157	2219*
2220	2234	2248	2250*	2251	2270*	2271	2285	2299	2301*	2302	4120	
990#	1493*	1494*	1495	1514*	1515*	1516	1540*	1541*	1542	1559*	1560*	1561
1580*	1581*	1582	1595*	1596*	1597	1612*	1613*	1614	1627*	1628*	1629	1699*
1700*	1701	1809*	1810*	1811	2382*	2383*	2384	2411*	2412*	2413	2426*	2427*
2428	2453*	2454*	2455	2576*								

CZRMHBO RMD3/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34

MACY11 30(1046) 21-NOV-77 14:15 PAGE 89
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0088

```

STYPDS      014616
STYPE       014150
STYPEC      014320
STYPEX      014366
STYPOC      014414
STYPON      014430
STYPOS      014370
$XTSTR      013472
$$GET4=     000000
$OFILL      014613
$40CAT=     ***** U
.           = 024432

```

[illegible]

SEQ 0089

AA	1872#	2017													
BB	1649#	1650	1760												
CHECK	581#	1484	1505	1529	1548	1571	1586	1603	1618	1690	1800	2373	2402	2417	2444
	2567	2596	2611	2638											
CLRATA	581#	2193													
COMMEN	716#														
ENDCOM	716#														
ERROR	610#	1437	1496	1517	1543	1562	1583	1598	1615	1630	1643	1702	1732	1745	1751
	1757	1812	1842	1855	1861	1867	1925	1938	1950	1962	1987	2002	2008	2014	2070
	2083	2095	2107	2132	2147	2153	2159	2226	2241	2247	2253	2277	2292	2298	2304
	2385	2414	2429	2456	2579	2608	2623	2650							
ESCAPE	716#														
FF	2311#	2312	2500												
GETPRI	716#														
GETSMR	582#	716#	1335												
HEAD	581#	1466	1666	1776	1889	2034	2182	2341	2535						
MORETA	945#	538													
MSG	1449#	1455	1650#	1653	1760#	1763	1872#	1876	2017#	2021	2164#	2167	2312#	2315	2500#
	2503														
MULT	716#														
NEUTRA	581#	1713	1823	1967	2112	2207	2258								
NEWTST	716#	1453	1651	1761	1874	2019	2165	2313	2501						
POP	716#	3145	3440	3489											
PUSH	716#	3104	3414	3469											
RELEAS	581#	1707	1817												
REPORT	716#														
SCOPE	611#	1647	1759	1869	2016	2161	2309	2498	2689						
SEIZE	581#	1910	2055												
SELECT	581#	1481	1502	1527	1546	1569	1601	1677	1710	1787	1820	1916	1927	1944	2061
	2072	2089	2352	2400	2438	2442	2476	2489	2546	2594	2632	2636	2668	2681	
SETATA	581#														
SETPRI	716#	3319													
SETTRA	3519#	3528	3529	3530	3531	3533	3535	3536	3537	3538	3539	3540			
SETUP	716#	1295													
SKIP	716#	2499													
SLASH	716#														
SPACE	716#														
STARS	716#	717	719	723	765	769	890	894	928	948	997	1025	1029	1285	1289
	1443	1447	1453	1465	1478	1524	1566	1634	1651	1665	1680	1687	1706	1761	1775
	1790	1797	1816	1874	1888	1903	1909	1941	1957	1964	2019	2033	2048	2054	2086
	2102	2109	2165	2181	2313	2340	2356	2364	2370	2389	2396	2435	2483	2501	2534
	2550	2558	2564	2583	2590	2629	2675	2694	2745	2749	2797	2803	2850	2891	2947
	3017	3094	3161	3230	3245	3308	3332	3400	3453	3498	3542	3546	3669	3673	4149
	4153														
SWITCH	581#														
SWRSU	716#	1315#													
TESTAG	581#	1476	1676	1786	1899	2044	2192	2351	2545						
TIMER	581#	1902	2047												
TRMTRP	3519#														
TYPBIN	716#														
TYPDEC	716#	2720	2727	2931											
TYPNAM	716#														
TYPNUM	716#														
TYPOCS	716#	1359	1365												
TYPOCT	716#	2903	2928	3258											
TYPTXT	716#	2716	2723												

M07

CZRMHB0 RM03/2 DU POR LGC 2
CZRMHB.P11 21-NOV-77 13:34MACY11 30(1046) 21-NOV-77 14:15 PAGE 92
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0090

USEREO	26928	2703										
SSCMRE	9468	985										
SSCNTM	9468	986	987	988	989	990						
SSSCA	7168											
SSNEWT	7168	1453	1651	1761	1874	2019	2165	2313	2501			
SSSET	35198	3528	3529	3530	3531	3533	3535	3536	3537	3538	3539	3540
SSSKIP	7168	2499										
.EQUAT	5828	606										
.HEADE	5828	583										
.SETUP	5828	1284										
.SUMMI	5828	594										
.SUMLO	5828	6048										
.SACTI	5828	926										
.SCATC	5828	916										
.SCMTA	5828	946										
.SEOP	5828	2692										
.SERRO	5828	2848										
.SERRT	5828	2889										
.SRDOC	5828	3398										
.SREAD	5828	3159										
.SSAVE	5828	3451										
.SSCOP	5828	2801										
.STRAP	5828	3496										
.STYPD	5828	3092										
.STYPE	5828	2945										
.STYPO	5828	3015										

. ABS. 024432 000

ERRORS DETECTED: 0

RM03:CZRMHB.BIN, RM03:CZRMHB.SEG/DOC/NL:TOC/SOL/CRF=RM03:CZRMHB.P11
 RUN-TIME: 20 13 1 SECONDS
 RUN-TIME RATIO: 468/35=13.1
 CORE USED: 28% (55 PAGES)

DOCUMENT PAGES: 90