

DA11F

BUS WINDOW STATIC TEST
MD-11-DZDAA-A

EP-DZDAA-A-DL-A
COPYRIGHT © 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN USA

The image displays a grid of 48 small tables, arranged in 8 rows and 6 columns. Each small table contains technical data, likely test results or component specifications, with various columns of numbers and text. The text is too small to read clearly but appears to be organized in a structured format. The tables are printed on a dark background, and the overall layout is consistent across the grid.

B01

MAINDEC-11-DZDAA-A
COPY 1.011

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 2

.REM %

MAINDEC-11-DZDAA-A
COPY 1.011
BUS WINDOW STATIC TEST
MACY11 27(732)
05-OCT-76 10:45
PAGE 2

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZDAA-A-D
PRODUCT NAME:	BUS WINDOW LOGIC TEST
DATE CREATED:	1-MAY-73
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	JOE STUBBLEBINE

COPYRIGHT (C) 1972
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS.

1

CONTENTS

- 1. ABSTRACT
- 2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
- 3. LOADING PROCEDURE
- 4. STARTING PROCEDURE
 - 4.1 CONTROL SWITCH SETTINGS
 - 4.2 STARTING ADDRESS
 - 4.3 PROGRAM AND/OR OPERATOR ACTION
- 5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.2 SUBROUTINE ABSTRACT
- 6. ERRORS
- 7. RESTRICTIONS
- 8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 POWER FAIL
- 9. PROGRAM DESCRIPTION

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

1. ABSTRACT

THIS PROGRAM TESTS THE LOGIC IN THE DA11-F BUS WINDOW. IT FIRST TESTS THE REGISTERS FOR WRITABILITY, THE CROSS COMMUNICATION BITS, AND RELOCATION REGISTERS. THERE ARE TWO DISTINCT SECTIONS. THE FIRST IS A 1 BUS TEST WHICH HAS BOTH SIDES OF THE WINDOW TIED INTO 1 BUS. THE SECOND IS THE 2 BUS TEST WHICH USES HANDSHAKING TO TEST ALL THE REGISTERS. THIS ALSO CHECKS THE WINDOW SIZE AND DOES TRANSFERS THROUGH THE WINDOW.

THE ONE BUS TEST IS PROVIDED FOR CONVENIENCE IN TROUBLE SHOOTING THE DA11-F MODULES. IT IS NOT USED IN THE NORMAL OPERATION OR MAINTENANCE OF A DUAL PROCESSOR SYSTEM. BEFORE USING THE ONE BUS TEST, REMOVE THE FOLLOWING WIRES FROM THE DA11-F WIRED ASSEMBLY BACKPLANE:

BUS A ACLO L: B01F1 TO B04F1

BUS A DCLO L: B01F2 TO B04F2

- AFTER TROUBLESHOOTING AN REPAIRING THE MODULES, REPLACE BOTH OF THESE WIRES BEFORE CONNECTING THE DA11-F INTO THE NORMAL DUAL PROCESSOR CONFIGURATION.

2. REQUIREMENTS

2.1 EQUIPMENT

ONE BUS TEST - ANY PDP-11 WITH 4K

TWO BUS TEST - ANY TWO PDP-11'S WITH 4K EACH

2.2 STORAGE

PROGRAM STORAGE - THE ROUTINES USE MEMORY 0 - 17776

2.3 PRELIMINARY PROGRAMS

THE BASIC PROCESSOR TESTS.

3. LOADING PROCEDURE: SEE 4.3

4. STARTING PROCEDURE

THERE ARE TWO WAYS OF STARTING THE PROGRAM. FIRST OF ALL

001-11-DZDAA-A
DZDAAA.P11
BUS WINDOW LOGIC TEST
DESCRIPTION

E01

MAINDEC-11-DZDAA-A
DZDAAA.P11

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 5

146
147

THE PROGRAM CAN BE LOADED INTO EACH PROCESSOR BY USING THE
ABS LOADER THE PROGRAMS CAN THEN BE STARTED AT THE STARTING

MAINDEC-11-DZDAA-A
DESCRIPTION

BUS WINDOW LOGIC TEST

PAGE 4

148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201

ADDRESS SPECIFIED IN SECTION 4.2 . THE OTHER OPTION IS TO LOAD THE PROGRAM INTO ONE PROCESSOR USING THE ABS LOADER AND THEN HAVE THE WINDOW TRANSFER THE PROGRAM TO THE OTHER PROCESSOR AS DESCRIBED IN SECTION 4.3 . TO USE THIS METHOD THE WINDOW MUST BE WORKING, IT MUST HAVE A SIZE OF AT LEAST 4K, AND THE WINDOW CANNOT BE LOCATED AT 32K OR ABOVE.

4.1 CONTROL SWITCH SETTINGS

SEE 5.1.1 (ALL DOWN FOR WORST CASE TESTING)

4.2 STARTING ADDRESSES

SA 200 TWO BUS TEST
SA 204 ONE BUS TEST
SA 210 POWER FAIL TEST - HOST - FIRST TO GO DOWN
SA 214 POWER FAIL TEST - NON HOST - SECOND TO GO DOWN
SA 220 POWER FAIL TEST - NEITHER HOST - FIRST TO GO DOWN
SA 224 POWER FAIL TEST - NEITHER HOST - SECOND TO GO DOWN
SA 510 TRANSFERS 0 - 17476 TO OTHER MACHINE (IF > 4K WINDOW)

4.3 PROGRAM AND/OR OPERATOR ACTION

- 1) LOAD THE ABS LOADER IF NOT IN MEMORY
- 2) LOAD PROGRAM WITH ABS LOADER
- 3) LOAD OTHER PROCESSOR IF TWO PROCESSOR TEST.
 - A) IF THE WINDOW IS NOT WORKING, USE ABS LOADER ON OTHER PROCESSOR
 - OR
 - B) IF THE WINDOW IS UP:
 - 1) LOAD ADDRESS 510
 - 2) START
 - 3) LOAD ADDRESS *500 IN OTHER PROCESSOR.
(* = STARTING ADDRESS OF THE WINDOW)
 - 4) START
 - 5) BOTH PROCESSORS WILL SELF START IF SWITCH 7 IS CLEAR. THE HOST WILL DELAY FOR ABOUT 30 SECONDS BEFORE STARTING.
 - 6) THE DISPLAY ON THE 11/45 WILL SHOW THE ITERATION COUNT IN THE LEFT BYTE AND TEST NUMBER IN THE RIGHT. TO USE, SET THE DATA DISPLAY SWITCH TO THE DISPLAY POSITION.

5. OPERATING PROCEDURE

MAINDEC-11-DZDAA-A
DESCRIPTION

BUS WINDOW LOGIC TEST

PAGE 5

202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257

5.1 OPERATIONAL SWITCH SETTINGS

AT SA 200 . . ALL SWITCHES DOWN IS WORST CASE TESTING FOR THE '2' BUS TEST. THE BELL WILL RING UPON COMPLETION OF A PASS.

5.1.1 SWITCH SETTINGS ARE:

SW<15> = 1 HALT ON ERROR
SW<14> = 1 LOOP ON TEST
SW<13> = 1 INHIBIT PRINTOUT
SW<10> = 1 BELL ON ERROR
 0 BELL ON PASS COMPLETE
SW<08> = 1 TYPE THE TEST NUMBER BEING EXECUTED
SW<07> = 1 IF SET, PREVENTS THE PROGRAM FROM SELF
 STARTING WHEN THE PROGRAM IS LOADED THROUGH
 THE WINDOW.

5.2 SUBROUTINE ABSTRACTS

5.2.1 SCOPE

THIS SUBROUTINE CALL IS PLACED BETWEEN EACH SUBTEST IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING ADDRESS OF EACH SUBTEST AS IT IS BEING ENTERED IN LOCATION "LAD". IF A SCOPE LOOP IS REQUESTED, THE CURRENT SUBTEST WILL BE LOOPED UPON. SW<11> ON A 1 INHIBITS ITERATION OF SUBTESTS. THE CONTENTS OF LAD MAY BE USED TO DETERMINE THE LAST SUBTEST SUCCESSFULLY COMPLETED.

5.2.2 HLT

THIS ROUTINE PRINTS OUT AN ERROR MESSAGE (SEE 6.1). TO INHIBIT TYPEOUTS, PUT SW<13> ON A 1.

5.2.3 TRAPCATCHER

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM 4 - 776 TO CATCH ANY UNEXPECTED TRAPS. THUS ANY UNEXPECTED TRAPS OR INTERRUPTS WILL HALT AT THE VECTOR + 2.

6. ERRORS

6.1 ERROR PRINTOUT

H01

MAINDEC-11-DZDAA-A
DZDAAA.P11

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 8

258

THE FORMAT IS AS FOLLOWS:

MAINDEC-11-DZDAA-A
DESCRIPTION

BUS WINDOW LOGIC TEST

PAGE 6

259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313

ADR RO RI

WHERE:

ADR = ADDRESS OF ERROR HLT
RO = BAD DATA
RI = GOOD DATA (OPTIONAL)

6.2 ERROR RECOVERY

ONE BUS TEST - RESTART AT 204

TWO BUS TEST - RESTART AT 200

7. RESTRICTIONS

WHEN RUNNING THE ONE BUS TEST, THE "B" SIDE OF THE WINDOW MUST HAVE ITS CSR REGISTER ADDRESS EXACTLY 20 GREATER THAN THE "A" SIDE CSR REGISTER. IF "A" SIDE CSR ADDRESS EQUALS 764000, THEN "B" SIDE CSR ADDRESS MUST BE 764020.

THE ONE BUS TEST IS PROVIDED FOR CONVENIENCE IN TROUBLE SHOOTING THE DA11-F MODULES. IT IS NOT USED IN THE NORMAL OPERATION OR MAINTENANCE OF A DUAL PROCESSOR SYSTEM. BEFORE USING THE ONE BUS TEST, REMOVE THE FOLLOWING WIRES FROM THE DA11-F WIRED ASSEMBLY BACKPLANE:

BUS A ACLO L: B01F1 TO B04F1

BUS A DCLO L: B01F2 TO B04F2

AFTER TROUBLESHOOTING AND REPAIRING THE MODULES, REPLACE BOTH OF THESE WIRES BEFORE CONNECTING THE DA11-F INTO THE NORMAL DUAL PROCESSOR CONFIGURATION.

8. MISCELLANEOUS

8.1 EXECUTION TIME

A BELL WILL RING WITHIN 1 MINUTE WITH ALL SWITCHES DOWN IN THE 2 BUS TEST.

8.2 STACK POINTER

STACK IS INITIALLY SET TO 500

314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369

MAINDEC-11-DZDAA-A
DESCRIPTION

BUS WINDOW LOGIC TEST

PAGE 7

8.3 POWER FAIL

THERE ARE SPECIAL ENTRY POINTS FOR POWER FAIL:

SA = 210 HOST COMPUTER - FIRST TO POWER DOWN AND UP
SA = 214 OTHER COMPUTER IS HOST - SECOND TO POWER DOWN AND UP
SA = 220 NEITHER COMPUTER IS HOST - FIRST TO POWER DOWN AND UP
SA = 224 NEITHER COMPUTER IS HOST - SECOND TO POWER DOWN AND UP

TO TEST POWER FAIL, START EACH MACHINE AT ITS APPROPRIATE ADDRESS (SEE ABOVE). THEN TURN THE FIRST MACHINE OFF AND BACK ON. THEN TURN THE SECOND MACHINE OFF AND BACK ON. WHEN FINISHED BOTH MACHINES SHOULD HALT. IF NEITHER MACHINE IS HOST, A THIRD POWER DOWN AND UP IS REQUIRED; I.E., POWER DOWN AND UP THE WINDOW POWER SUPPLY. HOST IS THE SOURCE OF POWER FOR THE BUS WINDOW.

9. PROGRAM DESCRIPTION

THIS PROGRAM IS A TEST OF THE DA11-F BUS WINDOW. IT HAS 2 INDEPENDANT SECTIONS. THE FIRST IS THE 1 BUS TEST WHICH HAS 1 UNIBUS TIED INTO BOTH SIDES OF THE WINDOW. THE REGISTER ADDRESSES AND TRAP VECTORS MUST BE DIFFERENT IN THIS TEST AND TRANSFERS THROUGH THE WINDOW ARE NOT ALLOWED BECAUSE OF THE DATIP CYCLE WOULD CAUSE A TIMEOUT. THIS SHOULD ONLY BE USED FOR INITIAL BRING UP OF THE WINDOW. THE SECOND PART OF THE PROGRAM IS FOR 2 BUSES AND 2 PROCESSORS. IT USES HANDSHAKING TO CHECK OUT THE REGISTERS. THIS SECTION IS SYNCHRONES AND IS IN 2 SECTIONS. THESE SECTIONS ARE SELECTED DEPENDING UPON WHO GETS THE BUS FIRST AND THEY SWITCH EVERY PASS. IN THIS TEST, TRANSFERS THROUGH THE WINDOW ARE TRIED INCLUDING A SECTION WHICH EXECUTES CODE IN THE OTHER PROCESSOR'S MEMORY.

THE ONE BUS TEST IS PROVIDED FOR CONVENIENCE IN TROUBLESHOOTING THE DA11-F MODULES. IT IS NOT USED IN THE NORMAL OPERATION OR MAINTENANCE OF A DUAL PROCESSOR SYSTEM. BEFORE USING THE ONE BUS TEST, REMOVE THE FOLLOWING WIRES FROM THE DA11-F WIRED ASSEMBLY BACKPLANE:

BUS A ACLO L: B01F1 TO B04F1

BUS A DCLO L: B01F2 TO B04F2

AFTER TROUBLESHOOTING AND REPAIRING THE MODULES, REPLACE BOTH OF THESE WIRES BEFORE CONNECTING THE DA11-F INTO THE NORMAL DUAL PROCESSOR CONFIGURATION.

%

.TITLE MAINDEC-11-DZDAA-A BUS WINDOW STATIC TEST
;COPYRIGHT 1972, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
;PROGRAM BY BOB BRAIN

370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407

	SWITCH	USE	
	----	-----	
	SW15= 100000	;HALT ON ERROR	
	SW14= 40000	;LOOP ON TEST	
	SW13= 20000	;INHIBIT ERROR TYPEOUTS	
	SW12= 10000		
	SW11= 4000	;INHIBIT ITERATIONS	
	SW10= 2000	;0 - BELL ON PASS COMPLETE	
		;1 - BELL ON ERROR	
		;HALT ON TRANSFER COMPLETE	
	BIT	WCSRA	WCSRB
	---	-----	-----
	15	A ERROR	B ERROR
	14	B TIME OUT	A TIME OUT
	13	B ACLO	A ACLO
	12	B NEW DATA	A NEW DATA
	11	B DATA 3	A DATA 3
	10	B DATA 2	A DATA 2
	9	B DATA 1	A DATA 1
	8	A TRANS ENABLE	B TRANS ENABLE
	7	B TRANS ENABLE	A TRANS ENABLE
	6	A I/E	B I/E
	5	A DATA 3	B DATA 3
	4	A DATA 2	B DATA 2
	3	A DATA 1	B DATA 1
	2	B WRITE ENABLE	A WRITE ENABLE
	1	A WRITE ENABLE	B WRITE ENABLE
	0	A NEW DATA	B NEW DATA

408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443

:LOADING PROCEDURE:

- ;1) LOAD THE ABS LOADER IF NOT IN MEMORY
- ;2) LOAD PROGRAM WITH ABS LOADER
- ;3) LOAD ADDRESS 510
- ;4) START
- ;5) LOAD ADDRESS *500 IN OTHER MACHINE (* = WINDOW ADDRESS)
- ;6) START

:TYPEOUTS:

;ADR RO R1

;ADR ADDRESS OF ERROR
 ;RO BAD DATA
 ;R1 GOOD DATA IF ANY (OPTIONAL)

000001
104400
104000
000004
177776
177570
177570
000007
000000
000001
000002
000003
000004
000005
000005
000006
000007

N= 1
 SCOPE= TRAP
 HLT= EMT
 TYPE= IOT
 PS= 177776
 SWR= 177570
 DISPLAY=SWR
 BELL= 7
 RO= %0
 R1= %1
 R2= %2
 R3= %3
 R4= %4
 R5= %5
 TTY= %5
 SP= %6
 PC= %7

MO1

MAINDEC-11-DZDAA-A
DZDAAA.P11

BJS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 13
SETUP AND EQUATS

```

444          000000          . = 0          ; TRAF CATCHER FROM 0 - 776
445
446          000200          . = 200
447
448 000200 000137 003410      JMP      @#BEGIN          ; JUMP TO BEGINING ADDRESS OF PROGRAM
449 000204 000137 001014      JMP      @#ONEBUS        ; START ADDRESS IF ON ONE BUS
450 000210 000137 010556      JMP      @#GO.HST        ; POWER FAIL - HOST
451 000214 000137 011060      JMP      @#GO.OTH        ; POWER FAIL - NON HOST
452 000220 000137 011352      JMP      @#GO.NE1       ; POWER FAIL - NEITHER HOST - FIRST
453 000224 000137 011776      JMP      @#GO.NE2       ; POWER FAIL - NEITHER HOST - SECOND
454
455          000500          . = 500
456
457 000500 000167 000010      JMP      GETCOR          ; MOVES POGRAM INTO OTHER MACHINE
458 000504 000000 000000      O,0
459 000510 000137 017440      JMP      @#OPEN          ; OPEN WINDOW AND WAIT
460
461 000514 010701          GETCOR: MOV      PC,R1          ; GET WINDOW ADDRESS
462 000516 000005          RESET          ; CLEAR THE WORLD
463 000520 162701 000516      SUB      @#GETCOR+2,R1    ; RELOCATE
464 000524 005000          CLR      R0              ; SETUP FOR TRANSFER
465 000526 012120          1$:      MOV      (1)+,(0)+      ; GET IT
466 000530 022700 017500      CMP      @#17500,R0      ; END?
467 000534 001374          BNE     1$              ; NO - LOOP
468 000536 032737 000200 177570 BIT      @#SW7,@#SWR      ; CHECK FOR HALT ON LOAD COMPLETE
469 000544 001401          BEQ     .+4             ; CONTINUE
470 000546 000000          HALT
471 000548 012767 000240 016720 MOV      @#NOP,BR.        ; MAKE IT EXECUTE A JMP @#BEGIN
472 000556 000137 003410      JMP      @#BEGIN          ; START THE TEST
473
474 000562 012700 164000          REGSCN: MOV      @#164000,R0    ; SET WCSRA FOR SCANNING
475 000566 012702 000010          MOV      @#8,R2          ; SET COUNT
476 000572 012737 000604 000004 MOV      @#5$,@#4        ; SET FOR TIMEOUT
477 000600 005710          6$:      TST      (0)            ; IS IT THERE?
478 000602 000207          RTS     PC              ; RETURN
479 000604 022626          5$:      CMP      (6)+,(6)+      ; CLEAR STACK
480 000606 062700 000020          ADD      @#20,R0         ; BUMP ADDRESS
481 000612 005302          DEC     R2              ; DEC COUNT
482 000614 001371          BNE     6$             ; LOOP UNTIL FOUND
483 000616 000000          HALT
484 000620 000760          BR     REGSCN          ; TRY IT AGAIN
485
486          000700          . = 700          ; EXECUTE FROM OTHER SIDE
487
488 000700 012701 000001          CODE1: MOV      @#1,%1      ; LOAD A BIT FOR SHIFTING
489 000704 010167 000030          MOV      @#1,CHKADR      ; LOAD THE ADDRESS
490 000710 000241          2$:      CLC
491 000712 006167 000022          ROL      CHKADR          ; CLEAR THE C BIT
492 000716 006101          ROL      %1              ; ROTATE IT
493 000720 103001          BCC     .+4             ; GET GOOD DATA
494 000722 000207          RTS     PC              ; SKIP IF MORE
495 000724 016700 000010          MOV      CHKADR,%0       ; RETURN TO OTHER SID.
496 000730 020001          CMP      %0,%1          ; GET FOR TYPING
497 000732 001401          BEQ     .+4             ; CHECK FOR BIT
498 000734 104001          HLT     +1              ; SKIP IF OK
499 000736 000764          BR     2$              ; ROTATE FAILED
                          ; LOOP

```

NO1

MAINDEC-11-DZDAA-A
DZDAAA.P11

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 14
SETUP AND EQUATS

500 000740 000000

CHKADR: 0

;ADDRESS OF BIT

```

501          001000          . =      1000
502
503
504 001000 000000          ICNT:  0          ;LH = ITERATION COUNT ;RH = TEST NO.
505 001002 000000          ERRORS: C          ;ERROR COUNT
506 001004 000000 000000 PCNT:  0,0         ;2 WORD PASS COUNT
507 001010 000000          LAD:  0          ;LOOP ADDRESS FOR SCOPE
508 001012 000000          HLTADR: 0          ;LAST HLT INSTRUCTION ADD. EXECUTED
509
510 001014 012706 000500 ONEBUS: MOV    #500,SP          ;SET SP ***** 500 *****
511 001020 004767 011604          JSR    PC,SETUP          ;SETUP THE VECTORS, ETC.
512 001024 012737 000340 177776 ONEB1: MOV    #340,2#PS        ;SET PRIO=7
513
514
515 *****
516 ;TEST 1          CHECK R/W OF WCSRA
517 *****
518
519 001032 104400          TST1:  SCOPE
520
521 001034 000005          RESET
522 001036 017700 013762          MOV    2WCSRA,RO          ;CLEAR THE WORLD
523 001042 022700 000000          CMP    #0,RO              ;GET THE DATA
524 001046 001401          BEQ    .+4                ;CHECK WCSRA FOR 0 ON RESET
525 001050 104000          HLT                      ;SKIP IF OK
526
527 001052 012777 000573 013744          MOV    #573,2WCSRA        ;SET THE SETABLE BITS
528 001060 017700 013740          MOV    2WCSRA,RO          ;GET THE RESULT
529 001064 022700 000573          CMP    #573!0!1,RO        ;CHECK FOR PROPER BITS
530 001070 001401          BEQ    .+4                ;SKIP IF OK
531 001072 104000          HLT                      ;WCSRA NOT 573!0
532
533 001074 012777 000452 013722          MOV    #452,2WCSRA        ;SET THE SETABLE BITS
534 001102 017700 013716          MOV    2WCSRA,RO          ;GET THE RESULT
535 001106 022700 000453          CMP    #452!0!1,RO        ;CHECK FOR PROPER BITS
536 001112 001401          BEQ    .+4                ;SKIP IF OK
537 001114 104000          HLT                      ;WCSRA NOT 452!0
538
539 001116 012777 000121 013700          MOV    #121,2WCSRA        ;SET THE SETABLE BITS
540 001124 017700 013674          MOV    2WCSRA,RO          ;GET THE RESULT
541 001130 022700 000121          CMP    #121!0!1,RO        ;CHECK FOR PROPER BITS
542 001134 001401          BEQ    .+4                ;SKIP IF OK
543 001136 104000          HLT                      ;WCSRA NOT 121!0
544
545 001140 005077 013660          CLR    2WCSRA              ;CLEAR ALL SETABLE BITS
546 001144 017700 013654          MOV    2WCSRA,RO          ;GET THE RESULT
547 001150 022700 000001          CMP    #0!1,RO            ;CHECK FOR PROPER BITS
548 001154 001401          BEQ    .+4                ;SKIP IF OK
549 001156 104000          HLT                      ;WCSRA NOT 0

```

548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581

001160 104400
001162 000005
001164 017700 013662
001170 022700 000000
001174 001401
001176 104000
001200 012777 000573 013644
001205 017700 013640
001212 022700 000573
001216 001401
001220 104000
001222 012777 000452 013622
001230 017700 013616
001234 022700 000453
001240 001401
001242 104000
001244 012777 000121 013600
001252 017700 013574
001256 022700 000121
001262 001401
001264 104000
001266 005077 013560
001272 017700 013554
001276 022700 000001
001302 001401
001304 104000

```
*****
:TEST 2          CHECK R/W OF WCSR8
*****
TST2:  SCOPE

RESET
MOV  @WCSR8,R0      ;CLEAR THE WORLD
CMP  #0,R0          ;GET THE DATA
BEQ  .+4            ;CHECK WCSR8 FOR 0 ON RESET
HLT  ;SKIP IF OK
      ;WCSR8 DID NOT RESET TO 0

MOV  #573,@WCSR8   ;SET THE SETABLE BITS
MOV  @WCSR8,R0     ;GET THE RESULT
CMP  #573!0!1,R0  ;CHECK FOR PROPER BITS
BEQ  .+4            ;SKIP IF OK
HLT  ;WCSR8 NOT 573!0

MOV  #452,@WCSR8   ;SET THE SETABLE BITS
MOV  @WCSR8,R0     ;GET THE RESULT
CMP  #452!0!1,R0  ;CHECK FOR PROPER BITS
BEQ  .+4            ;SKIP IF OK
HLT  ;WCSR8 NOT 452!0

MOV  #121,@WCSR8   ;SET THE SETABLE BITS
MOV  @WCSR8,R0     ;GET THE RESULT
CMP  #121!0!1,R0  ;CHECK FOR PROPER BITS
BEQ  .+4            ;SKIP IF OK
HLT  ;WCSR8 NOT 121!0

CLR  @WCSR8        ;CLEAR ALL SETABLE BITS
MOV  @WCSR8,R0     ;GET THE RESULT
CMP  #0!1,R0      ;CHECK FOR PROPER BITS
BEQ  .+4            ;SKIP IF OK
HLT  ;WCSR8 NOT 0
```



```

*****
:TEST 3          CHECK R/W OF WDBRA
*****
TST3:  SCOPE

585 001306 104000
586
587 001310 000005          RESET          ;CLEAR THE WORLD
588 001312 017700 013510  MOV @WDBRA,RO  ;GET THE DATA
589 001316 022700 000000  CMP #0,RO     ;CHECK WDBRA FOR 0 ON RESET
590 001322 001401          BEQ .+4       ;SKIP IF OK
591 001324 104000          HLT           ;WDBRA DID NOT RESET TO 0
592
593 001326 012777 177777 013472  MOV #-1,@WDBRA ;SET THE SETABLE BITS
594 001334 017700 013466  MOV @WDBRA,RO  ;GET THE RESULT
595 001340 022700 177777  CMP #-1!0,RO  ;CHECK FOR PROPER BITS
596 001344 001401          BEQ .+4       ;SKIP IF OK
597 001346 104000          HLT           ;WDBRA NOT -!0
598
599 001350 012777 125252 013450  MOV #125252,@WDBRA ;SET THE SETABLE BITS
600 001356 017700 013444  MOV @WDBRA,RO  ;GET THE RESULT
601 001362 022700 125252  CMP #125252!0,RO ;CHECK FOR PROPER BITS
602 001366 001401          BEQ .+4       ;SKIP IF OK
603 001370 104000          HLT           ;WDBRA NOT 125252!0
604
605 001372 012777 052525 013426  MOV #52525,@WDBRA ;SET THE SETABLE BITS
606 001400 017700 013422  MOV @WDBRA,RO  ;GET THE RESULT
607 001404 022700 052525  CMP #52525!0,RO ;CHECK FOR PROPER BITS
608 001410 001401          BEQ .+4       ;SKIP IF OK
609 001412 104000          HLT           ;WDBRA NOT 52525!0
610
611 001414 005077 013406  CLR @WDBRA     ;CLEAR ALL SETABLE BITS
612 001420 017700 013402  MOV @WDBRA,RO  ;GET THE RESULT
613 001424 022700 000000  CMP #0,RO     ;CHECK FOR PROPER BITS
614 001430 001401          BEQ .+4       ;SKIP IF OK
615 001432 104000          HLT           ;WDBRA NOT 0

```

```

616                                     :*****
617                                     :TEST 4          CHECK R/W OF WDBRBX
618                                     :*****
619 001434 104400                       †TST4:  SCOPE
620
621 001436 000005                       RESET
622 001440 017700 013410                MOV    @WDBRBX,RO      ;CLEAR THE WORLD
623 001444 022700 000000                CMP    #0,RO          ;GET THE DATA
624 001450 001401                       BEQ    .+4            ;CHECK WDBRBX FOR 0 ON RESET
625 001452 104000                       HLT
626                                     ;SKIP IF OK
627 001454 012777 177777 013372         MOV    #-1,@WDBRBX    ;WDBRBX DID NOT RESET TO 0
628 001462 017700 013366                 MOV    @WDBRBX,RO      ;SET THE SETABLE BITS
629 001466 022700 177777                 CMP    #-1!0,RO       ;GET THE RESULT
630 001472 001401                       BEQ    .+4            ;CHECK FOR PROPER BITS
631 001474 104000                       HLT
632                                     ;SKIP IF OK
633 001476 012777 125252 013350         MOV    #125252,@WDBRBX ;WDBRBX NOT -1!0
634 001504 017700 013344                 MOV    @WDBRBX,RO      ;SET THE SETABLE BITS
635 001510 022700 125252                 CMP    #125252!0,RO   ;GET THE RESULT
636 001514 001401                       BEQ    .+4            ;CHECK FOR PROPER BITS
637 001516 104000                       HLT
638                                     ;SKIP IF OK
639 001520 012777 052525 013326         MOV    #52525,@WDBRBX ;WDBRBX NOT 125252!0
640 001526 017700 013322                 MOV    @WDBRBX,RO      ;SET THE SETABLE BITS
641 001532 022700 052525                 CMP    #52525!0,RO    ;GET THE RESULT
642 001536 001401                       BEQ    .+4            ;CHECK FOR PROPER BITS
643 001540 104000                       HLT
644                                     ;SKIP IF OK
645 001542 005077 013306                 CLR    @WDBRBX         ;WDBRBX NOT 52525!0
646 001546 017700 013302                 MOV    @WDBRBX,RO      ;CLEAR ALL SETABLE BITS
647 001552 022700 000000                CMP    #0,RO          ;GET THE RESULT
648 001556 001401                       BEQ    .+4            ;CHECK FOR PROPER BITS
649 001560 104000                       HLT
649                                     ;SKIP IF OK
                                     ;WDBRBX NOT 0

```

```

650                                     ;*****
651                                     ;TEST 5      CHECK WDBRA TO WDBRAX CROSS COMMUNICATION
652                                     ;*****
653 001562 104400                       †TST5:  SCOPE
654
655 001564 000005                       RESET
656 001566 017700 013264                MOV    @WDBRAX,RO      ;CLEAR THE WORLD
657 001572 022700 000000                CMP    #0,RO          ;GET THE DATA
658 001576 001401                        BEQ    .+4             ;CHECK WDBRAX FOR 0 ON RESET
659 001600 104000                        HLT                               ;SKIP IF OK
660                                     ;WDBRAX DID NOT RESET TO 0
661 001602 012777 177777 013216        MOV    #-1,@WDBRA     ;SET THE SETABLE BITS
662 001610 017700 013242                MOV    @WDBRAX,RO     ;GET THE RESULT
663 001614 022700 177777                CMP    #-1!0,RO       ;CHECK FOR PROPER BITS
664 001620 001401                        BEQ    .+4             ;SKIP IF OK
665 001622 104000                        HLT                               ;WDBRAX NOT -1!0
666
667 001624 012777 125252 013174        MOV    #125252,@WDBRA ;SET THE SETABLE BITS
668 001632 017700 013220                MOV    @WDBRAX,RO     ;GET THE RESULT
669 001636 022700 125252                CMP    #125252!0,RO   ;CHECK FOR PROPER BITS
670 001642 001401                        BEQ    .+4             ;SKIP IF OK
671 001644 104000                        HLT                               ;WDBRAX NOT 125252!0
672
673 001646 012777 052525 013152        MOV    #52525,@WDBRA ;SET THE SETABLE BITS
674 001654 017700 013176                MOV    @WDBRAX,RO     ;GET THE RESULT
675 001660 022700 052525                CMP    #52525!0,RO    ;CHECK FOR PROPER BITS
676 001664 001401                        BEQ    .+4             ;SKIP IF OK
677 001666 104000                        HLT                               ;WDBRAX NOT 52525!0
678
679 001670 005077 013132                CLR    @WDBRA         ;CLEAR ALL SETABLE BITS
680 001674 017700 013156                MOV    @WDBRAX,RO     ;GET THE RESULT
681 001700 022700 000000                CMP    #0,RO          ;CHECK FOR PROPER BITS
682 001704 001401                        BEQ    .+4             ;SKIP IF OK
683 001706 104000                        HLT                               ;WDBRAX NOT 0

```

```

684 :*****
685 :TEST 6 CHECK WDBRBX TO WDBRB CROSS COMMUNICATION
686 :*****
687 001710 104400 TST6: SCOPE
688
689 001712 000005 RESET ;CLEAR THE WORLD
690 001714 017700 013110 MOV @WDBRB,RO ;GET THE DATA
691 001720 022700 000000 CMP #0,RO ;CHECK WDBRB FOR 0 ON RESET
692 001724 001401 BEQ .+4 ;SKIP IF OK
693 001726 104000 HLT ;WDBRB DID NOT RESET TO 0
694
695 001730 012777 177777 013116 MOV #-1,@WDBRBX ;SET THE SETABLE BITS
696 001736 017700 013066 MOV @WDBRB,RO ;GET THE RESULT
697 001742 022700 177777 CMP #-1!0,RO ;CHECK FOR PROPER BITS
698 001746 001401 BEQ .+4 ;SKIP IF OK
699 001750 104000 HLT ;WDBRB NOT -1!0
700
701 001752 012777 125252 013074 MOV #125252,@WDBRBX ;SET THE SETABLE BITS
702 001760 017700 013044 MOV @WDBRB,RO ;GET THE RESULT
703 001764 022700 125252 CMP #125252!0,RO ;CHECK FOR PROPER BITS
704 001770 001401 BEQ .+4 ;SKIP IF OK
705 001772 104000 HLT ;WDBRB NOT 125252!0
706
707 001774 012777 052525 013052 MOV #52525,@WDBRBX ;SET THE SETABLE BITS
708 002002 017700 013022 MOV @WDBRB,RO ;GET THE RESULT
709 002006 022700 052525 CMP #52525!0,RO ;CHECK FOR PROPER BITS
710 002012 001401 BEQ .+4 ;SKIP IF OK
711 002014 104000 HLT ;WDBRB NOT 52525!0
712
713 002016 005077 013032 CLR @WDBRBX ;CLEAR ALL SETABLE BITS
714 002022 017700 013002 MOV @WDBRB,RO ;GET THE RESULT
715 002026 022700 000000 CMP #0,RO ;CHECK FOR PROPER BITS
716 002032 001401 BEQ .+4 ;SKIP IF OK
717 002034 104000 HLT ;WDBRB NOT 0

```


H02

MAINDEC-11-DZDAA-A
DZDAAA.P11 TST6

BJS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 21
CHECK WDBRBX TO WDBRE CROSS COMMUNICATION

```
718 002036 016702 013036      MOV      SIZE,R2      ;GET PAGE SIZE
719 002042 005102              COM      R2          ;GET THE REVERSE
720
721 ;*****
722 ;TEST 7      CHECK R/W OF WRARA
723 ;*****
724 002044 104400      TST7:  SCOPE
725
726 002046 012703 000000      MOV      #0,R3      ;GET THE DATA
727 002052 040203      BIC      R2,R3      ;CLEAR THE UNUSED BITS
728 002054 010377 012754      MOV      R3,@WRARA  ;SET THE BITS IN WRARA
729 002060 017700 012750      MOV      @WRARA,R0  ;GET THE DATA
730 002064 040200      BIC      R2,R0      ;CLEAR THE JUNK
731 002066 020300      CMP      R3,R0      ;CHECK THE DATA
732 002070 001401      BEQ      .+4        ;SKIP IF OK
733 002072 104000      HLT
734
735 002074 012703 177777      MOV      #-1,R3     ;GET THE DATA
736 002100 040203      BIC      R2,R3     ;CLEAR THE UNUSED BITS
737 002102 010377 012726      MOV      R3,@WRARA ;SET THE BITS IN WRARA
738 002106 017700 012722      MOV      @WRARA,R0 ;GET THE DATA
739 002112 040200      BIC      R2,R0     ;CLEAR THE JUNK
740 002114 020300      CMP      R3,R0     ;CHECK THE DATA
741 002116 001401      BEQ      .+4        ;SKIP IF OK
742 002120 104000      HLT
743
744 002122 012703 052525      MOV      #52525,R3  ;GET THE DATA
745 002126 040203      BIC      R2,R3     ;CLEAR THE UNUSED BITS
746 002130 010377 012700      MOV      R3,@WRARA ;SET THE BITS IN WRARA
747 002134 017700 012674      MOV      @WRARA,R0 ;GET THE DATA
748 002140 040200      BIC      R2,R0     ;CLEAR THE JUNK
749 002142 020300      CMP      R3,R0     ;CHECK THE DATA
750 002144 001401      BEQ      .+4        ;SKIP IF OK
751 002146 104000      HLT
752
753 002150 012703 125252      MOV      #125252,R3 ;GET THE DATA
754 002154 040203      BIC      R2,R3     ;CLEAR THE UNUSED BITS
755 002156 010377 012652      MOV      R3,@WRARA ;SET THE BITS IN WRARA
756 002162 017700 012646      MOV      @WRARA,R0 ;GET THE DATA
757 002166 040200      BIC      R2,R0     ;CLEAR THE JUNK
758 002170 020300      CMP      R3,R0     ;CHECK THE DATA
759 002172 001401      BEQ      .+4        ;SKIP IF OK
760 002174 104000      HLT
761
762 ;*****
763 ;TEST 7      CHECK R/W OF WRARA
764 ;*****
```

```

761 :*****
762 :TEST 10 CHECK R/W OF WRARB
763 :*****
764 002176 104400 TST10: SCOPE
765
766 002200 012703 000000 MOV #0,R3 ;GET THE DATA
767 002204 040203 BIC R2,R3 ;CLEAR THE UNUSED BITS
768 002206 010377 012650 MOV R3,@WRARB ;SET THE BITS IN WRARB
769 002212 017700 012644 MOV @WRARB,R0 ;GET THE DATA
770 002216 040200 BIC R2,R0 ;CLEAR THE JUNK
771 002220 020300 CMP R3,R0 ;CHECK THE DATA
772 002222 001401 BEQ .+4 ;SKIP IF OK
773 002224 104000 HLT ;WRARB NOT 0
774
775 002226 012703 177777 MOV #-1,R3 ;GET THE DATA
776 002232 040203 BIC R2,R3 ;CLEAR THE UNUSED BITS
777 002234 010377 012622 MOV R3,@WRARB ;SET THE BITS IN WRARB
778 002240 017700 012616 MOV @WRARB,R0 ;GET THE DATA
779 002244 040200 BIC R2,R0 ;CLEAR THE JUNK
780 002246 020300 CMP R3,R0 ;CHECK THE DATA
781 002250 001401 BEQ .+4 ;SKIP IF OK
782 002252 104000 HLT ;WRARB NOT -1
783
784 002254 012703 125252 MOV #125252,R3 ;GET THE DATA
785 002260 040203 BIC R2,R3 ;CLEAR THE UNUSED BITS
786 002262 010377 012574 MOV R3,@WRARB ;SET THE BITS IN WRARB
787 002266 017700 012570 MOV @WRARB,R0 ;GET THE DATA
788 002272 040200 BIC R2,R0 ;CLEAR THE JUNK
789 002274 020300 CMP R3,R0 ;CHECK THE DATA
790 002276 001401 BEQ .+4 ;SKIP IF OK
791 002300 104000 HLT ;WRARB NOT 125252
792
793 002302 012703 052525 MOV #52525,R3 ;GET THE DATA
794 002306 040203 BIC R2,R3 ;CLEAR THE UNUSED BITS
795 002310 010377 012546 MOV R3,@WRARB ;SET THE BITS IN WRARB
796 002314 017700 012542 MOV @WRARB,R0 ;GET THE DATA
797 002320 040200 BIC R2,R0 ;CLEAR THE JUNK
798 002322 020300 CMP R3,R0 ;CHECK THE DATA
799 002324 001401 BEQ .+4 ;SKIP IF OK
800 002326 104000 HLT ;WRARB NOT 52525

```

```

*****
:TEST 11 CHECK CROSS COMMUNICATION BITS BETWEEN WCSRA AND WCSRB
*****
TST11: SCOPE

801
802
803
804 002330 104400
805
806 002332 012777 000000 012464 MOV #0,@WCSRA ;LOAD 0 INTO WCSRA
807 002340 017700 012506 012464 MOV @WCSRB,R0 ;SAVE DATA FOR TYPING
808 002344 022700 000000 CMP #0,R0 ;CHECK WCSRB FOR 0
809 002350 001401 BEQ .+4 ;SKIP IF OK
810 002352 104000 HLT ;DATA IN WCSRB NOT 0
811
812 002354 012777 000010 012442 MOV #10,@WCSRA ;LOAD 10 INTO WCSRA
813 002362 017700 012464 012464 MOV @WCSRB,R0 ;SAVE DATA FOR TYPING
814 002366 022700 001000 CMP #1000,R0 ;CHECK WCSRB FOR 1000
815 002372 001401 BEQ .+4 ;SKIP IF OK
816 002374 104000 HLT ;DATA IN WCSRB NOT 1000
817
818 002376 012777 000020 012420 MOV #20,@WCSRA ;LOAD 20 INTO WCSRA
819 002404 017700 012442 012442 MOV @WCSRB,R0 ;SAVE DATA FOR TYPING
820 002410 022700 002000 CMP #2000,R0 ;CHECK WCSRB FOR 2000
821 002414 001401 BEQ .+4 ;SKIP IF OK
822 002416 104000 HLT ;DATA IN WCSRB NOT 2000
823
824 002420 012777 000030 012376 MOV #30,@WCSRA ;LOAD 30 INTO WCSRA
825 002426 017700 012420 012420 MOV @WCSRB,R0 ;SAVE DATA FOR TYPING
826 002432 022700 003000 CMP #3000,R0 ;CHECK WCSRB FOR 3000
827 002436 001401 BEQ .+4 ;SKIP IF OK
828 002440 104000 HLT ;DATA IN WCSRB NOT 3000
829
830 002442 012777 000040 012354 MOV #40,@WCSRA ;LOAD 40 INTO WCSRA
831 002450 017700 012376 012376 MOV @WCSRB,R0 ;SAVE DATA FOR TYPING
832 002454 022700 004000 CMP #4000,R0 ;CHECK WCSRB FOR 4000
833 002460 001401 BEQ .+4 ;SKIP IF OK
834 002462 104000 HLT ;DATA IN WCSRB NOT 4000
835
836 002464 012777 000050 012332 MOV #50,@WCSRA ;LOAD 50 INTO WCSRA
837 002472 017700 012354 012354 MOV @WCSRB,R0 ;SAVE DATA FOR TYPING
838 002476 022700 005000 CMP #5000,R0 ;CHECK WCSRB FOR 5000
839 002502 001401 BEQ .+4 ;SKIP IF OK
840 002504 104000 HLT ;DATA IN WCSRB NOT 5000

```

K02

MAINDEC-11-DZDAA-A
DZDAAA.P11

TST11

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 24
CHECK CROSS COMMUNICATION BITS BETWEEN WCSRA AND WCSRB

841							
842	002506	012777	000060	012310	MOV	#60, @WCSRA	;LOAD 60 INTO WCSRA
843	002514	017700	012332		MOV	@WCSRB, R0	;SAVE DATA FOR TYPING
844	002520	022700	006000		CMP	#6000, R0	;CHECK WCSRB FOR 6000
845	002524	001401			BEQ	.+4	;SKIP IF OK
846	002526	104000			HLT		;DATA IN WCSRB NOT 6000
847							
848	002530	012777	000070	012266	MOV	#70, @WCSRA	;LOAD 70 INTO WCSRA
849	002536	017700	012310		MOV	@WCSRB, R0	;SAVE DATA FOR TYPING
850	002542	022700	007000		CMP	#7000, R0	;CHECK WCSRB FOR 7000
851	002546	001401			BEQ	.+4	;SKIP IF OK
852	002550	104000			HLT		;DATA IN WCSRB NOT 7000
853							
854	002552	012777	000002	012244	MOV	#2, @WCSRA	;LOAD 2 INTO WCSRA
855	002560	017700	012266		MOV	@WCSRB, R0	;SAVE DATA FOR TYPING
856	002564	022700	000004		CMP	#4, R0	;CHECK WCSRB FOR 4
857	002570	001401			BEQ	.+4	;SKIP IF OK
858	002572	104000			HLT		;DATA IN WCSRB NOT 4
859							
860	002574	012777	000400	012222	MOV	#400, @WCSRA	;LOAD 400 INTO WCSRA
861	002602	017700	012244		MOV	@WCSRB, R0	;SAVE DATA FOR TYPING
862	002606	022700	000200		CMP	#200, R0	;CHECK WCSRB FOR 200
863	002612	001401			BEQ	.+4	;SKIP IF OK
864	002614	104000			HLT		;DATA IN WCSRB NOT 200
865							
866	002616	012777	000001	012200	MOV	#1, @WCSRA	;LOAD 1 INTO WCSRA
867	002624	017700	012222		MOV	@WCSRB, R0	;SAVE DATA FOR TYPING
868	002630	022700	010000		CMP	#10000, R0	;CHECK WCSRB FOR 10000
869	002634	001401			BEQ	.+4	;SKIP IF OK
870	002636	104000			HLT		;DATA IN WCSRB NOT 10000
871							
872	002640	005077	012160		CLR	@WCSRA	;CLEAR THE
873	002644	005077	012202		CLR	@WCSRB	;REGISTER

L02

MAINDEC-11-DZDAA-A
DZDAAA.P11 TST12

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 25
CHECK CROSS COMMUNICATION BITS BETWEEN WCSR8 AND WCSRA

```

874                                     ;*****
875                                     ;TEST 12          CHECK CROSS COMMUNICATION BITS BETWEEN WCSR8 AND WCSRA
876                                     ;*****
877 002650 104400                       †TST12: SCOPE
878
879 002652 012777 000000 012172         MOV    #0,WCSR8           ;LOAD 0 INTO WCSR8
880 002660 017700 012140                 MOV    WCSR8,R0         ;SAVE DATA FOR TYPING
881 002664 022700 000000                 CMP    #0,R0           ;CHECK WCSRA FOR 0
882 002670 001401                         BEQ    .+4              ;SKIP IF OK
883 002672 104000                         HLT                    ;DATA IN WCSRA NOT 0
884
885 002674 012777 000010 012150         MOV    #10,WCSR8        ;LOAD 10 INTO WCSR8
886 002702 017700 012116                 MOV    WCSR8,R0         ;SAVE DATA FOR TYPING
887 002706 022700 001000                 CMP    #1000,R0        ;CHECK WCSRA FOR 1000
888 002712 001401                         BEQ    .+4              ;SKIP IF OK
889 002714 104000                         HLT                    ;DATA IN WCSRA NOT 1000
890
891 002716 012777 000020 012126         MOV    #20,WCSR8        ;LOAD 20 INTO WCSR8
892 002724 017700 012074                 MOV    WCSR8,R0         ;SAVE DATA FOR TYPING
893 002730 022700 002000                 CMP    #2000,R0        ;CHECK WCSRA FOR 2000
894 002734 001401                         BEQ    .+4              ;SKIP IF OK
895 002736 104000                         HLT                    ;DATA IN WCSRA NOT 2000
896
897 002740 012777 000030 012104         MOV    #30,WCSR8        ;LOAD 30 INTO WCSR8
898 002746 017700 012052                 MOV    WCSR8,R0         ;SAVE DATA FOR TYPING
899 002752 022700 003000                 CMP    #3000,R0        ;CHECK WCSRA FOR 3000
900 002756 001401                         BEQ    .+4              ;SKIP IF OK
901 002760 104000                         HLT                    ;DATA IN WCSRA NOT 3000
902
903 002762 012777 000040 012062         MOV    #40,WCSR8        ;LOAD 40 INTO WCSR8
904 002770 017700 012030                 MOV    WCSR8,R0         ;SAVE DATA FOR TYPING
905 002774 022700 004000                 CMP    #4000,R0        ;CHECK WCSRA FOR 4000
906 003000 001401                         BEQ    .+4              ;SKIP IF OK
907 003002 104000                         HLT                    ;DATA IN WCSRA NOT 4000
908
909 003004 012777 000050 012040         MOV    #50,WCSR8        ;LOAD 50 INTO WCSR8
910 003012 017700 012006                 MOV    WCSR8,R0         ;SAVE DATA FOR TYPING
911 003016 022700 005000                 CMP    #5000,R0        ;CHECK WCSRA FOR 5000
912 003022 001401                         BEQ    .+4              ;SKIP IF OK
913 003024 104000                         HLT                    ;DATA IN WCSRA NOT 5000

```

M02

MAINDEC-11-DZDAA-A
DZDAAA.P11

TST12

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 26
CHECK CROSS COMMUNICATION BITS BETWEEN WCSR8 AND WCSRA

914								
915	003026	012777	000060	012016	MOV	#60, @WCSR8		; LOAD 60 INTO WCSR8
916	003034	017700	011764		MOV	@WCSRA, R0		; SAVE DATA FOR TYPING
917	003040	022700	005000		CMP	#6000, R0		; CHECK WCSRA FOR 6000
918	003044	001401			BEQ	.+4		; SKIP IF OK
919	003046	104000			HLT			; DATA IN WCSRA NOT 6000
920								
921	003050	012777	000070	011774	MOV	#70, @WCSR8		; LOAD 70 INTO WCSR8
922	003056	017700	011742		MOV	@WCSRA, R0		; SAVE DATA FOR TYPING
923	003062	022700	007000		CMP	#7000, R0		; CHECK WCSRA FOR 7000
924	003066	001401			BEQ	.+4		; SKIP IF OK
925	003070	104000			HLT			; DATA IN WCSRA NOT 7000
926								
927	003072	012777	000002	011752	MOV	#2, @WCSR8		; LOAD 2 INTO WCSR8
928	003100	017700	011720		MOV	@WCSRA, R0		; SAVE DATA FOR TYPING
929	003104	022700	000004		CMP	#4, R0		; CHECK WCSRA FOR 4
930	003110	001401			BEQ	.+4		; SKIP IF OK
931	003112	104000			HLT			; DATA IN WCSRA NOT 4
932								
933	003114	012777	000400	011730	MOV	#400, @WCSR8		; LOAD 400 INTO WCSR8
934	003122	017700	011676		MOV	@WCSRA, R0		; SAVE DATA FOR TYPING
935	003126	022700	000200		CMP	#200, R0		; CHECK WCSRA FOR 200
936	003132	001401			BEQ	.+4		; SKIP IF OK
937	003134	104000			HLT			; DATA IN WCSRA NOT 200
938								
939	003136	012777	000001	011706	MOV	#1, @WCSR8		; LOAD 1 INTO WCSR8
940	003144	017700	011654		MOV	@WCSRA, R0		; SAVE DATA FOR TYPING
941	003150	022700	010000		CMP	#10000, R0		; CHECK WCSRA FOR 10000
942	003154	001401			BEQ	.+4		; SKIP IF OK
943	003156	104000			HLT			; DATA IN WCSRA NOT 10000
944								
945	003160	005077	011656		CLR	@WCSR8		; CLEAR THE
946	003164	005077	011634		CLR	@WCSRA		; REGISTER

```

947 ;*****
948 ;TEST 13 CHECK NEW DATA INTERUPT
949 ;*****
950 003170 104400 TST13: SCOPE
951
952 003172 005037 177776 CLR @#PS ;CLEAR PS
953 003176 012777 003314 011640 MOV #1$,@WVECA ;SET UP A VECTOR
954 003204 012777 003240 011660 MOV #2$,@WVECB ;SET UP B VECTOR
955 003212 012777 000100 011604 MOV #100,@WCSRA ;CLEAR A STATUS I/E
956 003220 012777 000100 011624 MOV #100,@WCSRB ;CLEAR B STATUS I/E
957 003226 052777 000001 011570 BIS #1,@WCSRA ;SET NEW DATA BIT ON A
958 003234 000240 NOP ;WAIT
959 003236 104000 HLT ;A SIDE SHOULD HAVE INTERUPTED
960 003240 012706 000500 2$: MOV #500,SP ;RESET SP
961 003244 017700 011602 MOV @WCSRB,RO ;GET B STATUS
962 003250 022700 010100 CMP #10100,RO ;CHECK B STATUS
963 003254 001401 BEQ .+4 ;SKIP IF OK
964 003256 104000 HLT ;B STATUS IS IS NOT 10100
965 003260 042777 010000 011564 BIC #10000,@WCSRE ;CLEAR THE BIT
966 003266 017700 011532 MOV @WCSRA,RO ;GET A STATUS
967 003272 022700 000100 CMP #100,RO ;CHECK A STATUS
968 003276 001401 BEQ .+4 ;SKIP IF OK
969 003300 104000 HLT ;A STATUS NOT 100
970
971 003302 052777 000001 011542 BIS #1,@WCSRB ;SET NEW DATA BIT ON B
972 003310 000240 NOP ;WAIT
973 003312 104000 HLT ;A SIDE SHOULD HAVE INTERUPTED
974 003314 012706 000500 1$: MOV #500,SP ;RESET SP
975 003320 017700 011500 MOV @WCSRA,RO ;GET A STATUS
976 003324 022700 010100 CMP #10100,RO ;CHECK A STATUS
977 003330 001401 BEQ .+4 ;SKIP IF OK
978 003332 104000 HLT ;A STATUS IS IS NOT 10100
979 003334 042777 010000 011462 BIC #10000,@WCSRA ;CLEAR THE BIT
980 003342 017700 011504 MOV @WCSRB,RO ;GET B STATUS
981 003346 022700 000100 CMP #100,RO ;CHECK B STATUS
982 003352 001401 BEQ .+4 ;SKIP IF OK
983 003354 104000 HLT ;B STATUS NOT 100
984
985 003356 005077 011442 CLR @WCSRA ;I/D
986 003362 005077 011464 CLR @WCSRB ;I/D
987
988
989 003366 104400 SCOPE ;LAST SCOPE
990 003370 032737 002000 177570 BIT #SW10,@#SWR ;RING BELL?
991 003376 001002 BNE 3$ ;SKIP IF SET
992 003400 000004 000007 TYPE BELL ;RING THE BELL
993 003404 000167 175404 3$: JMP ONEBUS ;LOOP ON ONEBUS TEST

```



```

994 003410 000005          BEGIN: RESET          ;CLEAR THE WORLD
995 003412 012706 000500      MOV      #500,SP          ;SET SP ***** 500 *****
996 003416 004767 007206      JSR      PC,SETUP        ;SETUP VECTORS, ETC.
997 003422 012737 000340 177776 GO:  MOV      #340,SPS          ;SET PRIO=7
998 003430 000240          NOP                    ;GIVE IT SOME ROOM
999 003432 032777 010000 011364 BIT      #10000,AWCSRA    ;CHECK FOR NEW DATA
1000 003440 001402          SEQ      .+6             ;SKIP IF FIRST ON THE BUS
1001 003442 000167 002752      JMP      SECOND          ;START SECOND SECTION
1002 003446 112767 000100 175324 MOVB    #100,ICNT        ;SET TEST 101

;*****
;TEST 101          CHECK R/W OF WCSRA
;*****
1003          TST101: SCOPE

1007 003454 104400

1009 003456 012777 000001 011340      MOV      #1,AWCSRA      ;CLEAR REGISTER
1010 003464 017702 011334      MOV      AWCSRA,R2     ;GET THE BITS
1011 003470 012777 000001 011326      MOV      #1,AWCSRA     ;SET THE NEW DATA BIT
1012 003476 017700 011322      MOV      AWCSRA,R0     ;GET FOR TYPING
1013 003502 040200          BIC      R2,R0          ;CLEAR THE JUNK
1014 003504 005700          TST      R0             ;CHECK FOR 0 BITS (1 MASKED)
1015 003506 001401          BEQ     .+4             ;SKIP IF OK
1016 003510 104000          HLT                    ;NEW DATA DID NOT SET

1019 003512 012703 000573      MOV      #573,R3        ;GET THE DATA
1020 003516 040203          BIC     R2,R3          ;CLEAR THE UNUSED BITS
1021 003520 010377 011300      MOV      R3,AWCSRA     ;SET THE BITS IN WCSRA
1022 003524 017700 011274      MOV      AWCSRA,R0     ;GET THE DATA
1023 003530 040200          BIC     R2,R0          ;CLEAR THE JUNK
1024 003532 020300          CMP     R3,R0          ;CHECK THE DATA
1025 003534 001401          BEQ     .+4             ;SKIP IF OK
1026 003536 104000          HLT                    ;WCSRA NOT 573

1029 003540 012703 000453      MOV      #453,R3        ;GET THE DATA
1030 003544 040203          BIC     R2,R3          ;CLEAR THE UNUSED BITS
1031 003546 010377 011252      MOV      R3,AWCSRA     ;SET THE BITS IN WCSRA
1032 003552 017700 011246      MOV      AWCSRA,R0     ;GET THE DATA
1033 003556 040203          BIC     R2,R0          ;CLEAR THE JUNK
1034 003560 020300          CMP     R3,R0          ;CHECK THE DATA
1035 003562 001401          BEQ     .+4             ;SKIP IF OK
1036 003564 104000          HLT                    ;WCSRA NOT 453

1039 003566 012703 000121      MOV      #121,R3        ;GET THE DATA
1040 003572 040203          BIC     R2,R3          ;CLEAR THE UNUSED BITS
1041 003574 010377 011224      MOV      R3,AWCSRA     ;SET THE BITS IN WCSRA
1042 003600 017700 011220      MOV      AWCSRA,R0     ;GET THE DATA
1043 003604 040200          BIC     R2,R0          ;CLEAR THE JUNK
1044 003606 020300          CMP     R3,R0          ;CHECK THE DATA
1045 003610 001401          BEQ     .+4             ;SKIP IF OK
1046 003612 104000          HLT                    ;WCSRA NOT 121
1047 003614 005077 011204      CLR     AWCSRA          ;ZERO THE REGISTER

```

1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075

003620 104400
003622 012777
003630 017700
003634 022700
003640 001401
003642 104000
003644 012777
003652 017700
003656 022700
003662 001401
003664 104000
003666 012777
003674 017700
003700 022700
003704 001401
003706 104000
003710 005077
003714 017700
003720 022700
003724 001401
003726 104000

177777 011176
011172
177777
052525 011154
011150
052525
125252 011132
011126
125252
011112
011106
000000

```
*****  
:TEST 102 CHECK R/W OF WDBRA  
*****  
TST102: SCOPE  
MOV #-1,WDBRA :SET THE SETABLE BITS  
MOV WDBRA,RO :GET THE RESULT  
CMP #-1!0,RO :CHECK FOR PROPER BITS  
BEQ .+4 :SKIP IF OK  
HLT :WDBRA NOT -1!0  
MOV #52525,WDBRA :SET THE SETABLE BITS  
MOV WDBRA,RO :GET THE RESULT  
CMP #52525!0,RO :CHECK FOR PROPER BITS  
BEQ .+4 :SKIP IF OK  
HLT :WDBRA NOT 52525!0  
MOV #125252,WDBRA :SET THE SETABLE BITS  
MOV WDBRA,RO :GET THE RESULT  
CMP #125252!0,RO :CHECK FOR PROPER BITS  
BEQ .+4 :SKIP IF OK  
HLT :WDBRA NOT 125252!0  
CLR WDBRA :CLEAR ALL SETABLE BITS  
MOV WDBRA,RO :GET THE RESULT  
CMP #0,RO :CHECK FOR PROPER BITS  
BEQ .+4 :SKIP IF OK  
HLT :WDBRA NOT 0
```

```

1076                                     :*****
1077                                     :TEST 103                               CHECK R/W OF WRARA
1078                                     :*****
1079 003730 104400                       TST103: SCOPE
1080
1081 003732 016702 011142                 MOV     SIZE,R2                       ;GET THE SIZE OF THE WINDOW
1082 003736 005102                       COM     R2                             ;SWITCH THE BITS
1083
1084 003740 012703 000000                 MOV     #0,R3                          ;GET THE DATA
1085 003744 040203                       BIC     R2,R3                          ;CLEAR THE UNUSED BITS
1086 003746 010377 011062                 MOV     R3,@WRARA                       ;SET THE BITS IN WRARA
1087 003752 017700 011056                 MOV     @WRARA,R0                       ;GET THE DATA
1088 003756 040200                       BIC     R2,R0                          ;CLEAR THE JUNK
1089 003760 020300                       CMP     R3,R0                          ;CHECK THE DATA
1090 003762 001401                       BEQ     .+4                             ;SKIP IF OK
1091 003764 104000                       HLT
                                           ;WRARA NOT 0
1092
1093 003766 012703 177777                 MOV     #-1,R3                         ;GET THE DATA
1094 003772 040203                       BIC     R2,R3                          ;CLEAR THE UNUSED BITS
1095 003774 010377 011034                 MOV     R3,@WRARA                       ;SET THE BITS IN WRARA
1096 004000 017700 011030                 MOV     @WRARA,R0                       ;GET THE DATA
1097 004004 040200                       BIC     R2,R0                          ;CLEAR THE JUNK
1098 004006 020300                       CMP     R3,R0                          ;CHECK THE DATA
1099 004010 001401                       BEQ     .+4                             ;SKIP IF OK
1100 004012 104000                       HLT
                                           ;WRARA NOT -1
1101
1102 004014 012703 125252                 MOV     #125252,R3                     ;GET THE DATA
1103 004020 040203                       BIC     R2,R3                          ;CLEAR THE UNUSED BITS
1104 004022 010377 011006                 MOV     R3,@WRARA                       ;SET THE BITS IN WRARA
1105 004026 017700 011002                 MOV     @WRARA,R0                       ;GET THE DATA
1106 004032 040200                       BIC     R2,R0                          ;CLEAR THE JUNK
1107 004034 020300                       CMP     R3,R0                          ;CHECK THE DATA
1108 004036 001401                       BEQ     .+4                             ;SKIP IF OK
1109 004040 104000                       HLT
                                           ;WRARA NOT 125252
1110
1111 004042 012703 052525                 MOV     #52525,R3                      ;GET THE DATA
1112 004046 040203                       BIC     R2,R3                          ;CLEAR THE UNUSED BITS
1113 004050 010377 010760                 MOV     R3,@WRARA                       ;SET THE BITS IN WRARA
1114 004054 017700 010754                 MOV     @WRARA,R0                       ;GET THE DATA
1115 004060 040200                       BIC     R2,R0                          ;CLEAR THE JUNK
1116 004062 020300                       CMP     R3,R0                          ;CHECK THE DATA
1117 004064 001401                       BEQ     .+4                             ;SKIP IF OK
1118 004066 104000                       HLT
                                           ;WRARA NOT 52525
1119 004070 005077 010740                 CLR     @WRARA                          ;CLEAR THE JUNK

```

E03

MAINDEC-11-DZDAA-A
DZDAAA.P11 TST104

BUS WINDOW STATIC TEST MACY!! 27(732) 05-OCT-76 10:45 PAGE 31
CHECK CROSS COMMUNICATION BITS

```

1120 :*****
1121 :TEST 104 CHECK CROSS COMMUNICATION BITS
1122 :*****
1123 004074 104400 TST104: SCOPE
1124
1125 004076 005037 000006 CLR 2#6 ;CLEAR THE STATUS WORD
1126 004102 005037 177776 CLR 2#PS ;SPL 0
1127 004106 032777 010000 010710 BIT #10000,2#WCSRA ;WAIT FOR NEW DATA
1128 004114 001774 BEQ -6 ;HANG
1129 004116 017700 010702 MOV 2#WCSRA,R0 ;GET 2#WCSRA FOR TYPING
1130 004122 022700 010000 CMP #10000,R0 ;IS 2#WCSRA = 10000?
1131 004126 001401 BEQ .+4 ;SKIP IF OK
1132 004130 104000 HLT ;2#WCSRA NOT 10000
1133
1134 004132 012777 000001 010664 MOV #0!1,2#WCSRA ;SET 0 INTO 2#WCSRA
1135 004140 004767 010214 JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
1136 004144 017700 010654 MOV 2#WCSRA,R0 ;GET 2#WCSRA FOR TYPING
1137 004150 022700 010000 CMP #10000,R0 ;IS 2#WCSRA = 10000?
1138 004154 001401 BEQ .+4 ;SKIP IF OK
1139 004156 104000 HLT ;2#WCSRA NOT 10000
1140
1141 004160 012777 000011 010636 MOV #10!1,2#WCSRA ;SET 10 INTO 2#WCSRA
1142 004166 004767 010166 JSR PC,TIMO ;TIME THE RESPONCE .. 2 ..
1143 004172 017700 010626 MOV 2#WCSRA,R0 ;GET 2#WCSRA FOR TYPING
1144 004176 022700 011010 CMP #11010,R0 ;IS 2#WCSRA = 11010?
1145 004202 001401 BEQ .+4 ;SKIP IF OK
1146 004204 104000 HLT ;2#WCSRA NOT 11010
1147
1148 004206 012777 000021 010610 MOV #20!1,2#WCSRA ;SET 20 INTO 2#WCSRA
1149 004214 004767 010140 JSR PC,TIMO ;TIME THE RESPONCE .. 3 ..
1150 004220 017700 010600 MOV 2#WCSRA,R0 ;GET 2#WCSRA FOR TYPING
1151 004224 022700 012020 CMP #12020,R0 ;IS 2#WCSRA = 12020?
1152 004230 001401 BEQ .+4 ;SKIP IF OK
1153 004232 104000 HLT ;2#WCSRA NOT 12020
1154
1155 004234 012777 000031 010562 MOV #30!1,2#WCSRA ;SET 30 INTO 2#WCSRA
1156 004242 004767 010112 JSR PC,TIMO ;TIME THE RESPONCE .. 4 ..
1157 004246 017700 010552 MOV 2#WCSRA,R0 ;GET 2#WCSRA FOR TYPING
1158 004252 022700 013030 CMP #13030,R0 ;IS 2#WCSRA = 13030?
1159 004256 001401 BEQ .+4 ;SKIP IF OK
1160 004260 104000 HLT ;2#WCSRA NOT 13030
1161
1162 004262 012777 000041 010534 MOV #40!1,2#WCSRA ;SET 40 INTO 2#WCSRA
1163 004270 004767 010064 JSR PC,TIMO ;TIME THE RESPONCE .. 5 ..
1164 004274 017700 010524 MOV 2#WCSRA,R0 ;GET 2#WCSRA FOR TYPING
1165 004300 022700 014040 CMP #14040,R0 ;IS 2#WCSRA = 14040?
1166 004304 001401 BEQ .+4 ;SKIP IF OK
1167 004306 104000 HLT ;2#WCSRA NOT 14040
1168

```

F03

MAINDEC-11-DZDAA-A BUS WINDOW STATIC TEST MACY!! 27(732) 05-OCT-76 10:45 PAGE 32
 DZDAAA.P11 TST104 CHECK CROSS COMMUNICATION BITS

1169	004310	012777	000051	010506	MOV	#50!1,@WCSRA	;SET 50 INTO @WCSRA	
1170	004316	004767	010036		JSR	PC,TIMO	;TIME THE RESPONCE	.. 6 ..
1171	004322	017700	010476		MOV	@WCSRA,R0	;GET @WCSRA FOR TYPING	
1172	004326	022700	015050		CMP	#15050,R0	;IS @WCSRA = 15050?	
1173	004332	001401			BEQ	+.4	;SKIP IF OK	
1174	004334	104000			HLT		;@WCSRA NOT 15050	
1175								
1176	004336	012777	000061	010460	MOV	#60!1,@WCSRA	;SET 60 INTO @WCSRA	
1177	004344	004767	010010		JSR	PC,TIMO	;TIME THE RESPONCE	.. 7 ..
1178	004350	017700	010450		MOV	@WCSRA,R0	;GET @WCSRA FOR TYPING	
1179	004354	022700	016060		CMP	#16060,R0	;IS @WCSRA = 16060?	
1180	004360	001401			BEQ	+.4	;SKIP IF OK	
1181	004362	104000			HLT		;@WCSRA NOT 16060	
1182								
1183	004364	012777	000071	010432	MOV	#70!1,@WCSRA	;SET 70 INTO @WCSRA	
1184	004372	004767	007762		JSR	PC,TIMO	;TIME THE RESPONCE	.. 10 ..
1185	004376	017700	010422		MOV	@WCSRA,R0	;GET @WCSRA FOR TYPING	
1186	004402	022700	017070		CMP	#17070,R0	;IS @WCSRA = 17070?	
1187	004406	001401			BEQ	+.4	;SKIP IF OK	
1188	004410	104000			HLT		;@WCSRA NOT 17070	
1189								
1190	004412	012777	000003	010404	MOV	#2!1,@WCSRA	;SET 2 INTO @WCSRA	
1191	004420	004767	007734		JSR	PC,TIMO	;TIME THE RESPONCE	.. 11 ..
1192	004424	017700	010374		MOV	@WCSRA,R0	;GET @WCSRA FOR TYPING	
1193	004430	022700	010006		CMP	#10006,R0	;IS @WCSRA = 10006?	
1194	004434	001401			BEQ	+.4	;SKIP IF OK	
1195	004436	104000			HLT		;@WCSRA NOT 10006	
1196								
1197	004440	012777	000401	010356	MOV	#400!1,@WCSRA	;SET 400 INTO @WCSRA	
1198	004446	004767	007706		JSR	PC,TIMO	;TIME THE RESPONCE	.. 12 ..
1199	004452	017700	010346		MOV	@WCSRA,R0	;GET @WCSRA FOR TYPING	
1200	004456	022700	010600		CMP	#10600,R0	;IS @WCSRA = 10600?	
1201	004462	001401			BEQ	+.4	;SKIP IF OK	
1202	004464	104000			HLT		;@WCSRA NOT 10600	
1203								
1204	004466	012777	000000	010330	MOV	#0,@WCSRA	;CHECK FOR BYTE ADDRESSABLE	
1205	004474	017700	010324		MOV	@WCSRA,R0	;GET @WCSRA FOR TYPING	
1206	004500	022700	010600		CMP	#10600,R0	;IS @WCSRA = 10600?	
1207	004504	001401			BEQ	+.4	;SKIP IF OK	
1208	004506	104000			HLT		;@WCSRA NOT 10600	

1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246

004510 104400
004512 012777 000001 010304
004520 004767 007634
004524 012777 177777 010274
004532 012777 000001 010264
004540 004767 007614
004544 017700 010260
004550 022700 177777
004554 001401
004556 104000
004560 012777 000000 010240
004566 012777 000001 010230
004574 004767 007560
004600 017700 010224
004604 022700 000000
004610 001401
004612 104000
004614 012777 052525 010204
004622 012777 000001 010174
004630 004767 007524
004634 017700 010170
004640 022700 052525
004644 001401
004646 104000
004650 012777 125252 010150
004656 012777 000001 010140
004664 004767 007470
004670 017700 010134
004674 022700 125252
004700 001401
004702 104000

```
*****
:TEST 105 CHECK CROSS WDBR'S WITH WINDOW CLOSED
*****
TST105: SCOPE
MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
MOV #-1,@WDBRA ;LOAD -1 INTO @WDBRA
MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 2 ..
MOV @WDBRB,RO ;GET @WDBRB FOR TYPING
CMP #-1,RO ;IS @WDBRB = -1?
BEQ .+4 ;SKIP IF OK
HLT ;@WDBRB NOT -1

MOV #0,@WDBRA ;LOAD 0 INTO @WDBRA
MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 3 ..
MOV @WDBRB,RO ;GET @WDBRB FOR TYPING
CMP #0,RO ;IS @WDBRB = 0?
BEQ .+4 ;SKIP IF OK
HLT ;@WDBRB NOT 0

MOV #52525,@WDBRA ;LOAD 52525 INTO @WDBRA
MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 4 ..
MOV @WDBRB,RO ;GET @WDBRB FOR TYPING
CMP #52525,RO ;IS @WDBRB = 52525?
BEQ .+4 ;SKIP IF OK
HLT ;@WDBRB NOT 52525

MOV #125252,@WDBRA ;LOAD 125252 INTO @WDBRA
MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 5 ..
MOV @WDBRB,RO ;GET @WDBRB FOR TYPING
CMP #125252,RO ;IS @WDBRB = 125252?
BEQ .+4 ;SKIP IF OK
HLT ;@WDBRB NOT 125252
```

```

1247
1248
1249
1250 004704 104400
1251
1252 004706 012777 000401 010110      MOV      #400!1,@WCSRA      ;SET 400 INTO @WCSRA
1253 004714 004767 007440                JSR      PC,TIM0          ;TIME THE RESPONCE      .. 1 ..
1254 004720 012777 052525 010100      MOV      #52525,@WDBRA   ;LOAD 52525 INTO @WDBRA
1255 004726 012777 000401 010070      MOV      #400!1,@WCSRA   ;SET 400 INTO @WCSRA
1256 004734 004767 007420                JSR      PC,TIM0          ;TIME THE RESPONCE      .. 2 ..
1257 004740 017700 010062                MOV      @WDBRA,R0       ;GET @WDBRA FOR TYPING
1258 004744 022700 125252                CMP      #125252,R0      ;IS @WDBRA = 125252?
1259 004750 001401                        BEQ      .+4              ;SKIP IF OK
1260 004752 104000                        HLT
1261
1262 004754 017700 010050                MOV      @WDBRB,R0       ;GET @WDBRB FOR TYPING
1263 004760 022700 125252                CMP      #125252,R0      ;IS @WDBRB = 125252?
1264 004764 001401                        BEQ      .+4              ;SKIP IF OK
1265 004766 104000                        HLT
1266
1267 004770 012777 000001 010026      MOV      #1!1,@WCSRA    ;SET 1 INTO @WCSRA
1268 004776 004767 007356                JSR      PC,TIM0          ;TIME THE RESPONCE      .. 3 ..
1269 005002 012777 052525 010016      MOV      #52525,@WDBRA  ;LOAD 52525 INTO @WDBRA
1270 005010 012777 000001 010006      MOV      #1!1,@WCSRA    ;SET 1 INTO @WCSRA
1271 005016 004767 007336                JSR      PC,TIM0          ;TIME THE RESPONCE      .. 4 ..
1272 005022 017700 010000                MOV      @WDBRA,R0       ;GET @WDBRA FOR TYPING
1273 005026 022700 052525                CMP      #52525,R0      ;IS @WDBRA = 52525?
1274 005032 001401                        BEQ      .+4              ;SKIP IF OK
1275 005034 104000                        HLT
1276
1277 005036 017700 007766                MOV      @WDBRB,R0       ;GET @WDBRB FOR TYPING
1278 005042 022700 052525                CMP      #52525,R0      ;IS @WDBRB = 52525?
1279 005046 001401                        BEQ      .+4              ;SKIP IF OK
1280 005050 104000                        HLT
1280

```



```

1281 :*****
1282 :TEST 107 CHECK ERROR CONDITIONS AND LOCKOUT
1283 :*****
1284 005052 104400 TST107: SCOPE
1285
1286 005054 012777 000001 007742 MOV #1!1, @WCSRA ;SET 1 INTO @WCSRA
1287 005062 004767 007272 JSR PC, TIME ;TIME THE RESPONCE .. 1 ..
1288 005066 012737 005104 000004 MOV #1$, @#4 ;SET FOR TIMEOUT
1289 005074 005777 007750 TST @WADRA ;TRY TO GO THROUGH A CLOSED WINDOW
1290 005100 104000 HLT ;NO TRAP WITH CLOSED WINDOW (BROKE IT)
1291 005102 000401 BR .+4 ;SKIP THE CLEAR
1292 005104 022626 1$: CMP (6)+, (6)+ ;CLEAR THE STACK
1293 005106 017700 007712 MOV @WCSRA, RO ;GET @WCSRA FOR TYPING
1294 005112 022700 110000 CMP #110000, RO ;IS @WCSRA = 110000?
1295 005116 001401 BEQ .+4 ;SKIP IF OK
1296 005120 104000 HLT ;@WCSRA NOT 110000
1297
1298 005122 012777 000001 007674 MOV #1!1, @WCSRA ;SET 1 INTO @WCSRA
1299 005130 004767 007224 JSR PC, TIME ;TIME THE RESPONCE .. 2 ..
1300 005134 012737 005150 000004 MOV #2$, @#4 ;SET FOR TIMEOUT
1301 005142 005777 007702 TST @WADRA ;NOW TRY THE WINDOW
1302 005146 000405 BR 3$ ;SHOULD NOT TRAP
1303 005150 017700 007650 2$: MOV @WCSRA, RO ;GET FOR TYPING
1304 005154 104000 HLT ;WINDOW TRAPPED ON DATAI
1305 005156 022626 CMP (6)+, (6)+ ;CLEAR THE STACK
1306 005160 000406 BR 4$ ;SKIP THE TEST
1307 005162 3$:
1308 005162 017700 007636 MOV @WCSRA, RO ;GET @WCSRA FOR TYPING
1309 005166 022700 010200 CMP #10200, RO ;IS @WCSRA = 10200?
1310 005172 001401 BEQ .+4 ;SKIP IF OK
1311 005174 104000 HLT ;@WCSRA NOT 10200
1312
1313 005176 012737 005212 000004 4$: MOV #5$, @#4 ;SET FOR TIMEOUT
1314 005204 005077 007640 CLR @WADRA ;TRY DATA0 WITH ONLY TRANSFER EN SET
1315 005210 000401 BR 14$ ;SKIP THE CLEAR
1316 005212 022626 5$: CMP (6)+, (6)+ ;CLEAR THE STACK
1317 005214 14$:
1318 005214 017700 007604 MOV @WCSRA, RO ;GET @WCSRA FOR TYPING
1319 005220 022700 110200 CMP #110200, RO ;IS @WCSRA = 110200?
1320 005224 001401 BEQ .+4 ;SKIP IF OK
1321 005226 104000 HLT ;@WCSRA NOT 110200
1322
1323 005230 012777 000001 007566 MOV #1!1, @WCSRA ;SET 1 INTO @WCSRA
1324 005236 004767 007116 JSR PC, TIME ;TIME THE RESPONCE .. 3 ..
1325 005242 012737 005260 000004 MOV #6$, @#4 ;SET FOR TIMEOUT
1326 005250 005077 007574 CLR @WADRA ;TRY DATA0 WITH ONLY WRITE ONLY SET
1327 005254 104000 HLT ;DID NOT TRAP ON ONLY WRITE ONLY
1328 005256 000401 BR .+4 ;SKIP CLEAR
1329 005260 022626 6$: CMP (6)+, (6)+ ;CLEAR THE JUNK
1330 005262 017700 007536 MOV @WCSRA, RO ;GET @WCSRA FOR TYPING
1331 005266 022700 110004 CMP #110004, RO ;IS @WCSRA = 110004?
1332 005272 001401 BEQ .+4 ;SKIP IF OK
1333 005274 104000 HLT ;@WCSRA NOT 110004
1334
1335 005276 012777 000001 007520 MOV #1!1, @WCSRA ;SET 1 INTO @WCSRA
1336 005304 004767 007050 JSR PC, TIME ;TIME THE RESPONCE .. 4 ..

```

1337	005310	012737	005324	000004		MOV	#7\$,a#4	;SET FOR TIMEOUT
1338	005316	025077	007526			CLR	WADRA	;NOW TRY IT WITH WINDOW OFEN
1339	005322	000405				BR	8\$;SKIP THE ERROR TYPEOUT
1340	005324	017700	007474		7\$:	MOV	WCSRA,RO	;GET FOR TYPING
1341	005330	104000				HLT		;TRAPPED WITH WINDOW OPEN
1342	005332	022626				CMP	(6)+,(6)+	;CLEAR THE STACK
1343	005334	000406				SR	9\$;SKIP THE TEST
1344	005336				8\$:			
1345	005336	017700	007462			MOV	WCSRA,RO	;GET WCSRA FOR TYPING
1346	005342	022700	010204			CMP	#10204,RO	;IS WCSRA = 10204?
1347	005346	001401				BEQ	+.4	;SKIP IF OK
1348	005350	104000				HLT		;WCSRA NOT 10204
1349								
1350	005352				9\$:			
1351	005352	012777	000001	007444		MOV	#1!1,WCSRA	;SET 1 INTO WCSRA
1352	005360	004767	006774			JSR	PC,TIMO	;TIME THE RESPONCE
1353	005364	005767	007506			TST	MM	;IS MEMORY MANAGEMENT IN USE? .. 5 ..
1354	005370	001403				BEQ	15\$;BRANCH IF NO
1355	005372	012701	020000			MOV	#020000,R1	;USE KIPARI FOR ADDR
1356	005376	000406				BR	16\$	
1357	005400	016701	007574		15\$:	MOV	OFFSET,R1	;GET THE OFFSET
1358	005404	016101	015142			MOV	ADR2(1),R1	;SET UP THE ADDRESS
1359	005410	066701	007434			ADD	WADRA,R1	;ADD IN WINDOW ADDRESS
1360	005414	012737	005426	000004	16\$:	MOV	#11\$,a#4	;SET FOR TIMEOUT
1361	005422	005011				CLR	(1)	;CAUSE A DATA0
1362	005424	000401				BR	+.4	;SKIP THE CLEAR
1363	005426	022626			11\$:	CMP	(6)+,(6)+	;CLEAR THE STACK
1364	005430	017700	007370			MOV	WCSRA,RO	;GET WCSRA FOR TYPING
1365	005434	022700	150204			CMP	#150204,RO	;IS WCSRA = 150204?
1366	005440	001401				BEQ	+.4	;SKIP IF OK
1367	005442	104000				HLT		;WCSRA NOT 150204
1368								
1369	005444	016700	007354			MOV	WCSRA,RO	;GET THE ADDRESS
1370	005450	142760	000200	000001		BICB	#200,1(0)	;CLEAR BIT 15
1371	005456	017700	007342			MOV	WCSRA,RO	;GET WCSRA FOR TYPING
1372	005462	022700	010204			CMP	#10204,RO	;IS WCSRA = 10204?
1373	005466	001401				BEQ	+.4	;SKIP IF OK
1374	005470	104000				HLT		;WCSRA NOT 10204
1375								
1376								
1377	005472	005767	007500			TST	MM	;MEMORY MANAGEMENT?
1378	005476	001405				BEQ	20\$;BRANCH IF NO
1379	005500	016701	007474			MOV	OFFSET,R1	
1380	005504	016101	015142			MOV	ADR2(1),R1	;GET EXPECTED WADRA CONTENTS
1381	005510	000402				BR	21\$	
1382	005512	166701	007332		20\$:	SUB	WADRA,R1	;GET RID OF THE BASE
1383	005516	017700	007310		21\$:	MOV	WADARA,RO	;GET FOR TYPING
1384	005522	020100				CMP	R1,RO	;CHECK WADARA FOR ERROR ADDRESS
1385	005524	001401				BEQ	+.4	;SKIP IF OK
1386	005526	104000				HLT		;THE WADARA CONTAINS THE WRONG ADDRESS

K03

```

1387                                     ;*****
1388                                     ;TEST 110 CHECK WINDOW SIZE AND WDARA
1389                                     ;*****
1390 005530 104400                       TST110. SCOPE
1391
1392 005532 005077 007276                 CLR QWRARA ;SET TO BANK 0
1393 005536 012777 000001 007260         MOV #1!1,QWCSRA ;SET 1 INTO QWCSRA
1394 005544 004767 006610                 JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
1395 005550 016702 007324                 MOV SIZE,R2 ;GET THE SIZE
1396 005554 005402                         NEG R2 ;MAKE
1397 005556 006302                         ASL R2 ;IT A
1398 005560 006302                         ASL R2 ;COUNT
1399 005562 032767 037777 007310         BIT #37777,SIZE ;IS IT 32K?
1400 005570 001002                         BNE 4$ ;SKIP IF NOT
1401 005572 012702 120000                 MOV #120000,R2 ;FUDGE FOR 32K
1402 005576 012737 005616 000004 4$:    MOV #2$,Q#4 ;SET FOR TIMEOUT
1403 005604 016701 007240                 MOV WADRA,R1 ;GET THE ADDRESS OF WINDOW
1404 005610 060102                         ADD R1,R2 ;CREATE END OF WINDOW
1405 005612 005011                         CLR (1) ;DO A DATA0
1406 005614 000401                         BR 3$ ;SKIP ERROR
1407 005616 022626                         2$:    CMP (6)+,(6)+ ;CLEAR THE STACK
1408 005620 017700 007200 3$:    MOV QWCSRA,R0 ;GET FOR TYPING
1409 005624 022700 110000                 CMP #110000,R0 ;ERROR FLAG SET?
1410 005630 001401                         BEQ .+4 ;SKIP IF OK
1411 005632 104000                         HLT ;NO ACCESS ERROR - BAD WONDOW SIZE?
1412 005634 042777 100000 007162         BIC #100000,QWCSRA ;CLEAR ERROR
1413 005642 166701 007202                 SUB WADRA,R1 ;WITH RESPECT TO WINDOW
1414 005646 017700 007160                 MOV QWDARA,R0 ;GET ADDRESS FOR TYPING
1415 005652 020001                         CMP R0,R1 ;CHECK ADDRESS
1416 005654 001401                         BEQ .+4 ;SKIP IF OK
1417 005656 104001                         HLT +1 ;ERROR ADDRESS IS WRONG
1418 005660 062701 000002                 ADD #2,R1 ;GO TO NEXT
1419 005664 066701 007160                 ADD WADRA,R1 ;MAKE IT WINDOW ADDRESS AGAIN
1420 005670 020102                         CMP R1,R2 ;CHECK FOR END
1421 005672 001347                         BNE 1$ ;LOOP UNTIL DONE
1422 005674 012737 000006 000004         MOV #6,Q#4 ;RESET 4
1423 005702 012777 000001 007114         MOV #1!1,QWCSRA ;SET 1 INTO QWCSRA
1424 005710 004767 006444                 JSR PC,TIMO ;TIME THE RESPONCE .. 2 ..

```

```

1425 ;*****
1426 ;TEST 111 CHECK WINDOW ADDRESSING CAPABILITY
1427 ;*****
1428 005714 104400 TST111: SCOPE
1429
1430 005716 005077 007112 CLR @WRARA ;SET TO BANK 0
1431 005722 012777 000001 007074 MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
1432 005730 004767 006424 JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
1433 005734 016702 007140 MOV SIZE,R2 ;GET THE SIZE
1434 005740 005402 NEG R2 ;MAKE IT A COUNT
1435 005742 006302 ASL R2
1436 005744 006302 ASL R2
1437 005746 032767 037777 007124 BIT #37777,SIZE ;IS IT 32K?
1438 005754 001002 BNE 4$ ;SKIP IF NOT
1439 005756 012702 120000 MOV #120000,R2 ;FUDGE FOR 32K
1440 005762 012737 006000 000004 4$: MOV #2$,@#4 ;SET FOR TIMEOUT
1441 005770 016701 007054 MOV WADRA,R1 ;GET WINDOW STARTING ADDR
1442 005774 011104 1$: MOV (R1),R4 ;GO THRU THE WINDOW
1443 005776 000401 BR 3$ ;SKIP IF ERROR
1444 006000 022626 2$: CMP (6)+,(6)+ ;RESTORE STACK
1445 006002 017700 007016 3$: MOV @WCSRA,R0 ;GET STATUS
1446 006006 022700 150204 CMP #150204,R0 ;CHECK STATUS
1447 006012 001401 BEQ .+4 ;SKIP IF OK
1448 006014 104000 HLT ;TIME OUT ERROR DID NOT OCCUR
1449 006016 042777 100000 007000 BIC #100000,@WCSRA ;CLEAR THE ERROR
1450 006024 012777 000001 006772 MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
1451 006032 004767 006322 JSR PC,TIMO ;TIME THE RESPONCE .. 2 ..
1452 006036 062701 000002 ADD #2,R1 ;UPDATE WINDOW ADDR
1453 006042 162702 000002 SUB #2,R2 ;DECREMENT SIZE COUNT
1454 006046 001352 BNE 1$ ;LOOP UNTIL DONE
1455 006050 012777 000071 006746 MOV #71!1,@WCSRA ;SET 71 INTO @WCSRA
1456 006056 004767 006276 JSR PC,TIMO ;TIME THE RESPONCE .. 3 ..
1457 ;*****
1458 ;TEST 112 CHECK OPEN WINDOW READ
1459 ;*****
1460 006062 104400 TST112: SCOPE
1461
1462 006064 012777 000001 006732 MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
1463 006072 004767 006262 JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
1464 006076 016700 006746 MOV WADRA,R0 ;GET WINDOW ADDRESS
1465 006102 062700 001014 ADD #ONEBUS,R0 ;START +ONEBUS
1466 006106 012703 001014 MOV #ONEBUS,R3 ;SET TO ADDRESS ONEBUS
1467 006112 016702 006762 MOV SIZE,R2 ;GET WINDOW SIZE
1468 006116 005402 NEG R2 ;MAKE IT THE LAST ADDRESS
1469 006120 006302 ASL R2 ;GET INTO
1470 006122 006302 ASL R2 ;POSITION
1471 006124 066702 006720 ADD WADRA,R2 ;RELOCATE TO BEGINNING OF WINDOW
1472 006130 011001 1$: MOV (0),R1 ;GET THE BAD DATA
1473 006132 021013 CMP (0),(3) ;CHECK THE DATA
1474 006134 001401 BEQ .+4 ;SKIP IF SAME
1475 006136 104001 HLT +1 ;WINDOW DATA NOT BANK 0 DATA
1476 006140 022320 CMP (3)+,(0)+ ;ADD 2 TO BOTH
1477 006142 022703 010330 CMP #DONE,R3 ;END OF DATA?
1478 006146 001402 BEQ 2$ ;YES - GET OUT
1479 006150 021200 CMP R2,R0 ;END OF WINDOW?
1480 006152 001366 BNE 1$ ;NO - LOOP

```

M03

1481 006154 2\$:
1482 006154 012777 000001 006642 MOV #1!1, @WCSRA ;SET 1 INTO @WCSRA
1483 006162 004767 006172 JSR PC, TIMO ;TIME THE RESPONCE .. 2 ..

1484
1485
1486 ;*****
1487 ;TEST 113 CHECK OPEN WINDOW WRITE
1488 ;*****
1489 006166 104400 †TST113: SCOPE

1491 006170 012777 000001 006626 MOV #1!1, @WCSRA ;SET 1 INTO @WCSRA
1492 006176 004767 006156 JSR PC, TIMO ;TIME THE RESPONCE .. 1 ..
1493 006202 016700 006642 MOV WADRA, R0 ;GET WINDOW ADDRESS
1494 006206 062700 000500 ADD #500, R0 ;START AT 500
1495 006212 012703 000100 MOV #100, R3 ;SET COUNT (500-676)
1496 006216 010320 1\$: MOV R3, (0)+ ;LOAD THE DATA
1497 006220 005303 DEC R3 ;DEC THE COUNT
1498 006222 001375 BNE 1\$;LOOP UNTIL DONE
1499 006224 162700 000200 SUB #200, R0 ;BACK IT UP
1500 006230 012703 000100 MOV #100, R3 ;SET COUNT
1501 006234 011001 2\$: MOV (0), R1 ;GET DATA FOR TYPING
1502 006236 021003 CMP (0), R3 ;CHECK DATA
1503 006240 001401 BEQ .+4 ;SKIP IF OK
1504 006242 104001 HLT +1 ;DATA IS WRONG
1505 006244 005720 TST (0)+ ;ADD 2
1506 006246 005303 DEC R3 ;DEC THE COUNT
1507 006250 001371 BNE 2\$;LOOP UNTIL END

```

1508 ;*****
1509 ;TEST 114 CHECK INSTRUCTION EXECUTION
1510 ;*****
1511 006252 104400 †TST114: SCOPE
1512
1513 006254 012777 000001 006542 MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
1514 006262 004767 006072 JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
1515 006266 016700 006556 MOV WADRA,RO ;GET WINDOW ADDRESS
1516 006272 004760 000700 JSR PC,CODE1(0) ;EXECUTE CODE IN OTHER MACHINE
1517
1518 ;*****
1519 ;TEST 115 CHECK INTERUPTS
1520 ;*****
1521 006276 104400 †TST115: SCOPE
1522
1523 006300 012777 006332 006536 MOV #4$,@WVECA ;SET FOR TRAP
1524 006306 013777 177776 006532 MOV @#PS,@WVECA+2 ;SET SAME PS
1525 006314 012777 000101 006502 MOV #100!1,@WCSRA ;SET 100 INTO @WCSRA
1526 006322 004767 006032 JSR PC,TIMO ;TIME THE RESPONCE .. i ..
1527 006326 104000 HLT ;B DID NOT INTERUPT A
1528 006330 000401 BR .+4 ;SKIP ON ERROR
1529 006332 022626 4$: CMP (6)+,(6)+ ;CLEAR STACK
1530 006334 005726 TST (6)+ ;CLEAR THE REST
1531 006336 017700 006462 MOV @WCSRA,RO ;GET @WCSRA FOR TYPING
1532 006342 022700 010100 CMP #10100,RO ;IS @WCSRA = 10100?
1533 006346 001401 BEQ .+4 ;SKIP IF OK
1534 006350 104000 HLT ;@WCSRA NOT 10100
1535 006352 016777 006470 006464 MOV WVECA+2,@WVECA ;RESET FOR
1536 006360 005077 006462 CLR @WVECA+2 ;ILL TRAP
1537
1538
1539 006364 012777 000001 006432 MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
1540 006372 004767 005762 JSR PC,TIMO ;TIME THE RESPONCE .. 2 ..
1541
1542 006376 012701 000045 20$: MOV #4$,R1 ;SET FOR DOUBLE LOOP
1543 006402 005000 10$: CLR RO ;WAIT
1544 006404 005300 DEC RO ;BEFORE
1545 006406 001376 BNE 10$ ;RESTARTING
1546 006410 005301 DEC R1 ;DEC COUNT
1547 006412 001373 BNE 20$ ;LOOP UNTIL DONE
1548
1549 006414 000167 001710 JMP DONE ;LOOP

```

```
1550 006420 112767 000200 172352 SECOND: MOVB #200,ICNT ;SET TEST 201
1551
1552 ;*****
1553 ;TEST 201 CHECK R/W OF WCSRA
1554 ;*****
1555 006426 104400 †TST201: SCOPE
1556
1557 006430 012777 010000 006365 MOV #10000,@WCSRA ;LOAD 10000 INTO @WCSRA
1558 006436 017700 006362 MOV @WCSRA,R0 ;GET FOR TYPING
1559 006442 022700 010000 CMP #10000,R0 ;CHECK @WCSRA FOR 10000
1560 006446 001401 BEQ .+4 ;SKIP IF OK
1561 006450 104000 HLT ;@WCSRA DOES NOT CONTAIN 10000
1562
1563 006452 012777 010572 006344 MOV #10572,@WCSRA ;LOAD 10572 INTO @WCSRA
1564 006460 017700 006340 MOV @WCSRA,R0 ;GET FOR TYPING
1565 006464 022700 010572 CMP #10572,R0 ;CHECK @WCSRA FOR 10572
1566 006470 001401 BEQ .+4 ;SKIP IF OK
1567 006472 104000 HLT ;@WCSRA DOES NOT CONTAIN 10572
1568
1569 006474 012777 010452 006322 MOV #10452,@WCSRA ;LOAD 10452 INTO @WCSRA
1570 006502 017700 006316 MOV @WCSRA,R0 ;GET FOR TYPING
1571 006506 022700 010452 CMP #10452,R0 ;CHECK @WCSRA FOR 10452
1572 006512 001401 BEQ .+4 ;SKIP IF OK
1573 006514 104000 HLT ;@WCSRA DOES NOT CONTAIN 10452
1574
1575 006516 012777 010120 006300 MOV #10120,@WCSRA ;LOAD 10120 INTO @WCSRA
1576 006524 017700 006274 MOV @WCSRA,R0 ;GET FOR TYPING
1577 006530 022700 010120 CMP #10120,R0 ;CHECK @WCSRA FOR 10120
1578 006534 001401 BEQ .+4 ;SKIP IF OK
1579 006536 104000 HLT ;@WCSRA DOES NOT CONTAIN 10120
1580 006540 012777 010000 006256 MOV #10000,@WCSRA ;ZERO THE REGISTER
```



```

1581
1582
1583
1584 006546 104400
1585
1586 006550 012777 177777 006250 MOV #-1,@WDBRA ;SET THE SETABLE BITS
1587 006556 017700 006244 MOV @WDBRA,R0 ;GET THE RESULT
1588 006562 022700 177777 CMP #-1!0,R0 ;CHECK FOR PROPER BITS
1589 006566 001401 BEQ .+4 ;SKIP IF OK
1590 006570 104000 HLT ;WDBRA NOT -1!0
1591
1592 006572 012777 052525 006226 MOV #52525,@WDBRA ;SET THE SETABLE BITS
1593 006600 017700 006222 MOV @WDBRA,R0 ;GET THE RESULT
1594 006604 022700 052525 CMP #52525!0,R0 ;CHECK FOR PROPER BITS
1595 006610 001401 BEQ .+4 ;SKIP IF OK
1596 006612 104000 HLT ;WDBRA NOT 52525!0
1597
1598 006614 012777 125252 006204 MOV #125252,@WDBRA ;SET THE SETABLE BITS
1599 006622 017700 006200 MOV @WDBRA,R0 ;GET THE RESULT
1600 006626 022700 125252 CMP #125252!0,R0 ;CHECK FOR PROPER BITS
1601 006632 001401 BEQ .+4 ;SKIP IF OK
1602 006634 104000 HLT ;WDBRA NOT 125252!0
1603
1604 006636 005077 006164 CLR @WDBRA ;CLEAR ALL SETABLE BITS
1605 006642 017700 006160 MOV @WDBRA,R0 ;GET THE RESULT
1606 006646 022700 000000 CMP #0,R0 ;CHECK FOR PROPER BITS
1607 006652 001401 BEQ .+4 ;SKIP IF OK
1608 006654 104000 HLT ;WDBRA NOT 0

```

```

:*****
;TEST 202 CHECK R/W OF WDBRA
:*****
†TST202: SCOPE

```

```

1609                                     :*****
1610                                     :TEST 203                CHECK R/W OF WRARA
1611                                     :*****
1612 006656 104400                       †TST203: SCOPE
1613
1614 006660 016702 006214                MOV    SIZE,R2          ;GET WINDOW SIZE
1615 006664 005102                       COM    R2               ;MAKE IT A MASK
1616
1617 006666 012703 000000                MOV    #0,R3           ;GET THE DATA
1618 006672 040203                       BIC    R2,R3           ;CLEAR THE UNUSED BITS
1619 006674 010377 006134                MOV    R3,@WRARA      ;SET THE BITS IN WRARA
1620 006700 017700 006130                MOV    @WRARA,R0      ;GET THE DATA
1621 006704 040200                       BIC    R2,R0           ;CLEAR THE JUNK
1622 006706 020300                       CMP    R3,R0           ;CHECK THE DATA
1623 006710 001401                       BEQ    .+4             ;SKIP IF OK
1624 006712 104000                       HLT                    ;WRARA NOT 0
1625
1626 006714 012703 177777                MOV    #-1,R3         ;GET THE DATA
1627 006720 040203                       BIC    R2,R3           ;CLEAR THE UNUSED BITS
1628 006722 010377 006106                MOV    R3,@WRARA      ;SET THE BITS IN WRARA
1629 006726 017700 006102                MOV    @WRARA,R0      ;GET THE DATA
1630 006732 040200                       BIC    R2,R0           ;CLEAR THE JUNK
1631 006734 020300                       CMP    R3,R0           ;CHECK THE DATA
1632 006736 001401                       BEQ    .+4             ;SKIP IF OK
1633 006740 104000                       HLT                    ;WRARA NOT -1
1634
1635 006742 012703 052525                MOV    #52525,R3      ;GET THE DATA
1636 006746 040203                       BIC    R2,R3           ;CLEAR THE UNUSED BITS
1637 006750 010377 006060                MOV    R3,@WRARA      ;SET THE BITS IN WRARA
1638 006754 017700 006054                MOV    @WRARA,R0      ;GET THE DATA
1639 006760 040200                       BIC    R2,R0           ;CLEAR THE JUNK
1640 006762 020300                       CMP    R3,R0           ;CHECK THE DATA
1641 006764 001401                       BEQ    .+4             ;SKIP IF OK
1642 006766 104000                       HLT                    ;WRARA NOT 52525
1643
1644 006770 012703 125252                MOV    #125252,R3     ;GET THE DATA
1645 006774 040203                       BIC    R2,R3           ;CLEAR THE UNUSED BITS
1646 006776 010377 006032                MOV    R3,@WRARA      ;SET THE BITS IN WRARA
1647 007002 017700 006026                MOV    @WRARA,R0      ;GET THE DATA
1648 007006 040200                       BIC    R2,R0           ;CLEAR THE JUNK
1649 007010 020300                       CMP    R3,R0           ;CHECK THE DATA
1650 007012 001401                       BEQ    .+4             ;SKIP IF OK
1651 007014 104000                       HLT                    ;WRARA NOT 125252
1652 007016 005077 006012                CLR    @WRARA         ;CLEAR JUNK

```

```

1653                                     ;*****
1654                                     ;TEST 204                                CHECK CROSS COMMUNICATION BITS
1655                                     ;*****
1656 007022 104400                         †TST204: SCOPE
1657
1658 007024 005037 000006                   CLR      @#6           ;CLEAR THE STATUS WORD
1659 007030 005037 177776                   CLR      @#PS         ;SPL 0
1660 007034 012777 000001 005762          MOV      @!1,@WCSRA   ;SET 1 INTO @WCSRA
1661 007042 004767 005312                   JSR      PC,TIMO      ;TIME THE RESPONCE      .. 1 ..
1662
1663 007046 017700 005752                   MOV      @WCSRA,RO    ;GET @WCSRA FOR TYPING
1664 007052 022700 010000                   CMP      @10000,RO    ;IS @WCSRA = 10000?
1665 007056 001401                          BEQ      .+4          ;SKIP IF OK
1666 007060 104000                          HLT
1667 007062 012777 000001 005734          MOV      @0!1,@WCSRA  ;SET 0 INTO @WCSRA
1668 007070 004767 005264                   JSR      PC,TIMO      ;TIME THE RESPONCE      .. 2 ..
1669
1670 007074 017700 005724                   MOV      @WCSRA,RO    ;GET @WCSRA FOR TYPING
1671 007100 022700 011000                   CMP      @11000,RO    ;IS @WCSRA = 11000?
1672 007104 001401                          BEQ      .+4          ;SKIP IF OK
1673 007106 104000                          HLT
1674 007110 012777 000011 005706          MOV      @10!1,@WCSRA ;SET 10 INTO @WCSRA
1675 007116 004767 005236                   JSR      PC,TIMO      ;TIME THE RESPONCE      .. 3 ..
1676
1677 007122 017700 005676                   MOV      @WCSRA,RO    ;GET @WCSRA FOR TYPING
1678 007126 022700 012010                   CMP      @12010,RO    ;IS @WCSRA = 12010?
1679 007132 001401                          BEQ      .+4          ;SKIP IF OK
1680 007134 104000                          HLT
1681 007136 012777 000021 005660          MOV      @20!1,@WCSRA ;SET 20 INTO @WCSRA
1682 007144 004767 005210                   JSR      PC,TIMO      ;TIME THE RESPONCE      .. 4 ..
1683
1684 007150 017700 005650                   MOV      @WCSRA,RO    ;GET @WCSRA FOR TYPING
1685 007154 022700 013020                   CMP      @13020,RO    ;IS @WCSRA = 13020?
1686 007160 001401                          BEQ      .+4          ;SKIP IF OK
1687 007162 104000                          HLT
1688 007164 012777 000031 005632          MOV      @30!1,@WCSRA ;SET 30 INTO @WCSRA
1689 007172 004767 005162                   JSR      PC,TIMO      ;TIME THE RESPONCE      .. 5 ..
1690
1691 007176 017700 005622                   MOV      @WCSRA,RO    ;GET @WCSRA FOR TYPING
1692 007202 022700 014030                   CMP      @14030,RO    ;IS @WCSRA = 14030?
1693 007206 001401                          BEQ      .+4          ;SKIP IF OK
1694 007210 104000                          HLT
1695 007212 012777 000041 005604          MOV      @40!1,@WCSRA ;SET 40 INTO @WCSRA
1696 007220 004767 005134                   JSR      PC,TIMO      ;TIME THE RESPONCE      .. 6 ..
1697

```

F04

MAINDEC-11-DZDAA-A BJS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 45
 DZDAAA.P11 TST204 CHECK CROSS COMMUNICATION BITS

1698	007224	017700	005574		MOV	QWCSRA,RO	;GET QWCSRA FOR TYPING	
1699	007230	022700	015040		CMP	#15040,RO	;IS QWCSRA = 15040?	
1700	007234	001401			BEQ	.+4	;SKIP IF OK	
1701	007236	104000			HLT		;QWCSRA NOT 15040	
1702	007240	012777	000051	005556	MOV	#50!1,QWCSRA	;SET 50 INTO QWCSRA	
1703	007246	004767	005106		JSR	PC,TIMO	;TIME THE RESPONCE	.. 7 ..
1704								
1705	007252	017700	005546		MOV	QWCSRA,RO	;GET QWCSRA FOR TYPING	
1706	007256	022700	016050		CMP	#16050,RO	;IS QWCSRA = 16050?	
1707	007262	001401			BEQ	.+4	;SKIP IF OK	
1708	007264	104000			HLT		;QWCSRA NOT 16050	
1709	007266	012777	000061	005530	MOV	#60!1,QWCSRA	;SET 60 INTO QWCSRA	
1710	007274	004767	005060		JSR	PC,TIMO	;TIME THE RESPONCE	.. 10 ..
1711								
1712	007300	017700	005520		MOV	QWCSRA,RO	;GET QWCSRA FOR TYPING	
1713	007304	022700	017060		CMP	#17060,RO	;IS QWCSRA = 17060?	
1714	007310	001401			BEQ	.+4	;SKIP IF OK	
1715	007312	104000			HLT		;QWCSRA NOT 17060	
1716	007314	012777	000071	005502	MOV	#70!1,QWCSRA	;SET 70 INTO QWCSRA	
1717	007322	004767	005032		JSR	PC,TIMO	;TIME THE RESPONCE	.. 11 ..
1718								
1719	007326	017700	005472		MOV	QWCSRA,RO	;GET QWCSRA FOR TYPING	
1720	007332	022700	010074		CMP	#10074,RO	;IS QWCSRA = 10074?	
1721	007336	001401			BEQ	.+4	;SKIP IF OK	
1722	007340	104000			HLT		;QWCSRA NOT 10074	
1723	007342	012777	000003	005454	MOV	#2!1,QWCSRA	;SET 2 INTO QWCSRA	
1724	007350	004767	005004		JSR	PC,TIMO	;TIME THE RESPONCE	.. 12 ..
1725								
1726	007354	017700	005444		MOV	QWCSRA,RO	;GET QWCSRA FOR TYPING	
1727	007360	022700	010202		CMP	#10202,RO	;IS QWCSRA = 10202?	
1728	007364	001401			BEQ	.+4	;SKIP IF OK	
1729	007366	104000			HLT		;QWCSRA NOT 10202	
1730	007370	012777	000401	005426	MOV	#400!1,QWCSRA	;SET 400 INTO QWCSRA	

1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769

007376 104400
007400 004767 004754
007404 012777 000001 005412
007412 004767 004742
007416 017700 005406
007422 022700 177777
007426 001401
007430 104000
007432 012777 177777 005366
007440 012777 000001 005356
007446 004767 004706
007452 017700 005352
007456 022700 000000
007462 001401
007464 104000
007466 012777 000000 005332
007474 012777 000001 005322
007502 004767 004652
007506 017700 005316
007512 022700 052525
007516 001401
007520 104000
007522 012777 052525 005276
007530 012777 000001 005266
007536 004767 004616
007542 017700 005262
007546 022700 125252
007552 001401
007554 104000
007556 012777 125252 005242
007564 012777 000001 005232

:TEST 205 CHECK CROSS WDBR'S WITH WINDOW CLOSED

TST205: SCOPE

JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
MOV #1!,@WCSRA ;SET 1 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 2 ..
MOV @WDBRB,R0 ;GET @WDBRB FOR TYPING
CMP #-1,R0 ;IS @WDBRB = -1?
BEQ .+4 ;SKIP IF OK
HLT ;@WDBRB NOT -1

MOV #-1,@WDBRA ;LOAD -1 INTO @WDBRA
MOV #1!,@WCSRA ;SET 1 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 3 ..
MOV @WDBRB,R0 ;GET @WDBRB FOR TYPING
CMP #0,R0 ;IS @WDBRB = 0?
BEQ .+4 ;SKIP IF OK
HLT ;@WDBRB NOT 0

MOV #0,@WDBRA ;LOAD 0 INTO @WDBRA
MOV #1!,@WCSRA ;SET 1 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 4 ..
MOV @WDBRB,R0 ;GET @WDBRB FOR TYPING
CMP #52525,R0 ;IS @WDBRB = 52525?
BEQ .+4 ;SKIP IF OK
HLT ;@WDBRB NOT 52525

MOV #52525,@WDBRA ;LOAD 52525 INTO @WDBRA
MOV #1!,@WCSRA ;SET 1 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 5 ..
MOV @WDBRB,R0 ;GET @WDBRB FOR TYPING
CMP #125252,R0 ;IS @WDBRB = 125252?
BEQ .+4 ;SKIP IF OK
HLT ;@WDBRB NOT 125252

MOV #125252,@WDBRA ;LOAD 125252 INTO WDBRA
MOV #1!,@WCSRA ;SET 1 INTO @WCSRA

```

1770 ;*****
1771 ;TEST 206 CHECK CROSS WDBR'S WITH WINDOW OPEN AND CLOSED
1772 ;*****
1773 007572 104400 †TST206: SCOPE
1774
1775 007574 004767 004560 JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
1776 007600 012777 000401 005216 MOV #400!1,@WCSRA ;SET 400 INTO @WCSRA
1777 007606 004767 004546 JSR PC,TIMO ;TIME THE RESPONCE .. 2 ..
1778 007612 017700 005212 MOV @WDBRB,R0 ;GET @WDBRB FOR TYPING
1779 007616 022700 125252 CMP #125252,R0 ;IS @WDBRB = 125252?
1780 007622 001401 BEQ .+4 ;SKIP IF OK
1781 007624 104000 HLT ;@WDBRB NOT 125252
1782
1783 007626 012777 052525 005172 MOV #52525,@WDBRA ;LOAD 52525 INTO @WDBRA
1784 007634 017700 005166 MOV @WDBRA,R0 ;GET @WDBRA FOR TYPING
1785 007640 022700 125252 CMP #125252,R0 ;IS @WDBRA = 125252?
1786 007644 001401 BEQ .+4 ;SKIP IF OK
1787 007646 104000 HLT ;@WDBRA NOT 125252
1788
1789 007650 012777 000001 005145 MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
1790 007656 004767 004476 JSR PC,TIMO ;TIME THE RESPONCE .. 3 ..
1791 007662 012777 052525 005136 MOV #52525,@WDBRA ;LOAD 52525 INTO @WDBRA
1792 007670 012777 000001 005126 MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
1793 007676 004767 004456 JSR PC,TIMO ;TIME THE RESPONCE .. 4 ..
1794 007702 017700 005120 MOV @WDBRA,R0 ;GET @WDBRA FOR TYPING
1795 007706 022700 052525 CMP #52525,R0 ;IS @WDBRA = 52525?
1796 007712 001401 BEQ .+4 ;SKIP IF OK
1797 007714 104000 HLT ;@WDBRA NOT 52525
1798 007716 012777 000001 005100 MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA

```

```

1799
1800
1801
1802 007724 104400
1803
1804 007726 004767 004426
1805 007732 005077 005076
1806 007736 012777 000001 005060
1807 007744 004767 004410
1808 007750 012777 000401 005046
1809 007756 004767 004376
1810 007762 012777 000003 005034
1811 007770 004767 004364
1812 007774 012777 000403 005022
1813 010002 004767 004352
1814 010006 016705 005166
1815 010012 016577 015124 005014
1816 010020 012777 000403 004776

```

```

:*****
:TEST 207 CHECK ERROR CONDITIONS AND LOCKOUT
:*****
†TST207: SCOPE
JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
CLR @WRARA ;MAP TO OK BANK
MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 2 ..
MOV #400!1,@WCSRA ;SET 400 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 3 ..
MOV #2!1,@WCSRA ;SET 2 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 4 ..
MOV #402!1,@WCSRA ;SET 402 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 5 ..
MOV OFFSET,TTY ;GET OFFSET
MOV ADRI(TTY),@WRARA ;SET TO PERF PAGE
MOV #402!1,@WCSRA ;SET 402 INTO @WCSRA

```

```

1817
1818
1819
1820
1821
1822 10026 104400
1823
1824 010030 004767 004324
1825 010034 005077 004774
1826 010040 012777 000001 004756
1827 010046 032777 010000 004750
1828 010054 001774
1829 010056 012777 000001 004740
1830
1831

```

```

:*****
:TEST 210 CHECK WINDOW SIZE AND WDARA
:*****
†TST210: SCOPE
JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
CLR @WRARA ;MAP TO 0
MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
BIT #10000,@WCSRA ;NEW DATA SET?
BEQ -5 ;HANG
MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA

```

```

1832
1833
1834
1835 010064 104400
1836
1837 010066 004767 004266
1838 010072 005067 005004
1839 010076 017777 004734 004730
1840 010104
1841 010104 012777 000403 004712
1842 010112 004767 004242
1843 010116 032777 007000 004700
1844 010124 001024
1845 010126 017700 004672
1846 010132 022700 110402
1847 010136 001401
1848 010140 104000
1849 010142 042777 100000 004654
1850 010150 017700 004656
1851 010154 016701 004722
1852 010160 020001
1853 010162 001401
1854 010164 104001

```

```

:*****
:TEST 211 CHECK WINDOW ADDRESSING CAPABILITY
:*****
†TST211: SCOPE
JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
CLR DISP ;INITIALIZE EXP DISPLACEMENT ADDR
MOV @WADRAX,@WRARA ;FORCE THE WINDOW TO RELOCATE TO ITSELF
15:
MOV #402!1,@WCSRA ;SET 402 INTO @WCSRA
JSR PC,TIMO ;TIME THE RESPONCE .. 2 ..
BIT #7000,@WCSRA ;IS TEST COMPLETE?
BNE 2$ ;BRANCH IF YES
MOV @WCSRA,R0 ;GET STATUS
CMP #110402,R0 ;IS IT CORRECT?
BEQ +4 ;SKIP IF OK
HLT ;NO TIMEOUT ERROR
BIC #100000,@WCSRA ;CLEAR THE ERROR
MOV @WDARA,R0 ;GET DISPLACEMENT ADDR
MOV DISP,R1 ;GET EXP DISPLACEMENT ADDR
CMP R0,R1 ;IS DISPLACEMENT ADDR CORRECT?
BEQ +4 ;SKIP IF OK
HLT +1 ;DISPLACEMENT ADDR INCORRECT

```


J04

1855 010166 062767 000002 004706 AUD #2,DISP ;UPDATE EXPECTED ADDR
1856 010174 000743 BR 15 ;LOOP
1857 010176 2\$:
1858 010176 012777 000001 004620 MOV #!1,@WCSRA ;SET 1 INTO @WCSRA
1859
1860

1861 ;*****
1862 ;TEST 212 CHECK OPEN WINDOW READ
1863 ;*****
1864 010204 104400 †TST212: SCOPE

1865
1866 010206 004767 004146 JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
1867 010212 005077 004616 CLR @WRARA ;SET TO BANK 0
1868 010216 012777 000403 004600 MOV #402!1,@WCSRA ;SET 402 INTO @WCSRA
1869 010224 032777 010000 004572 BIT #1000,@WCSRA ;NEW DATA SET?
1870 010232 001774 BEQ -6 ;HANG
1871 010234 012777 000001 004562 MOV #!1,@WCSRA ;SET 1 INTO @WCSRA
1872
1873

1874 ;*****
1875 ;TEST 213 CHECK OPEN WINDOW WRITE
1876 ;*****
1877 010242 104400 †TST213: SCOPE

1878
1879 010244 004767 004110 JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
1880 010250 005077 004560 CLR @WRARA ;SET TO BANK 0
1881 010254 012777 000403 004542 MOV #402!1,@WCSRA ;SET 402 INTO @WCSRA

```

1882 ;*****
1883 ;TEST 214 CHECK INSTRUCTION EXECUTION
1884 ;*****
1895 010262 104400 †TST214: SCOPE
1886
1887 010264 004767 004070 JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
1888
1889 010270 005077 004540 CLR @WRARA ;SET TO BANK 0
1890 010274 012777 000403 004522 MOV #402!1,@WCSRA ;SET 402 INTO @WCSRA
1891
1892 ;*****
1893 ;TEST 215 CHECK INTERUPTS
1894 ;*****
1895 010302 104400 †TST215: SCOPE
1896
1897 010304 004767 004050 JSR PC,TIMO ;TIME THE RESPONCE .. 1 ..
1898 010310 012777 000001 004506 MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA
1899
1900 010316 004767 004036 JSR PC,TIMO ;TIME THE RESPONCE .. 2 ..
1901 010322 012777 000001 004474 MOV #1!1,@WCSRA ;SET 1 INTO @WCSRA

```

1902	010330	104400			DONE:	SCOPE		
1903	010332	062767	000001	170446		ADD	#1,PCNT+2	;ADD ONE TO THE PASS COUNT
1904	010340	005567	170440			ADC	PCNT	;MAKE IT DOUBLE PREC.
1905	010344	032737	002000	177570		BIT	#SW10,@#SWR	;RING THE BELL?
1906	010352	001002				BNE	4\$;NO
1907	010354	000004	000007			TYPE	BELL	;RING THE BELL
1908	010360	013700	000042		4\$:	MOV	@#42,R0	;GET MONITOR ADDRESS
1909	010364	001404				BEQ	3\$;BRANCH IF NONE
1910	010366	004710				JSR	7,(0)	;GO TO MONITOR
1911	010370	000240				NOP		
1912	010372	000240				NOP		
1913	010374	000240				NOP		
1914	010376	000167	173020		3\$:	JMP	GO	;RETURN
1915								
1916								
1917								
1918								
1919								
1920								
1921								
1922								
1923								
1924								
1925								
1926								
1927	010402	010467	000146		IOTS:	MOV	R4,IOTS	;SAVE R4
1928	010406	010546				MOV	TTY,-(6)	;SAVE TTY R5=TTY
1929	010410	017605	000002			MOV	@2(6),TTY	;GET ADDRESS TO BE TYPED
1930	010414	032705	177400			BIT	#177400,TTY	;IS IT A TYPED?
1931	010420	001004				BNE	1\$;NO
1932	010422	010567	000124			MOV	TTY,TYPE	;GET THE CHARACTER
1933	010426	012705	010552			MOV	#.TYPE,TTY	;FUJGE THE ADDRESS
1934	010432	105715			1\$:	TSTB	(TTY)	;TERMINATOR?
1935	010434	001423				BEQ	2\$;GET OUT IF SO
1936	010436	122715	000012			CMPB	#12,(TTY)	;IS THE CHAR A LINE FEED?
1937	010442	001012				BNE	4\$;BRANCH IF NO
1938	010444	116704	000101			MOVB	FILCHR+1,R4	;GET FILLER COUNT
1939	010450	116737	000074	177566	5\$:	MOVB	FILCHR,@#177566	;OUTPUT FILLER CHAR
1940	010456	105737	177564			TSTB	@#177564	;WAIT FOR DONE
1941	010462	100375				BP	-4	
1942	010464	005304				DEC	R4	;DECREMENT FILLER COUNT
1943	010466	001370				BNE	5\$	
1944	010470	112537	177566		4\$:	MOVB	(TTY)+,@#177566	;LOAD AND TYPE THE CHARACTER
1945	010474	105737	177564			TSTB	@#177564	;IS THE PRINTER READY
1946	010500	100375				BPL	-4	;WAIT UNTIL IT IS
1947	010502	000753				BR	1\$;GET THE NEXT CHARACTER
1948	010504	017646	000002		2\$:	MOV	@2(6),-(6)	;GET ADDRESS TO BE TYPED
1949	010510	062766	000002	000004		ADD	#2,4(6)	;ADD 2 TO THE ADDRESS
1950	010516	022666	000002			CMP	(6)+,2(6)	;IS IT .+2?
1951	010522	001006				BNE	3\$;NO
1952	010524	062705	000002			ADD	#2,TTY	;ADD 2 TO THE ADDRESS
1953	010530	042705	000001			BIC	#1,TTY	;BACK UP TO AN EVEN BYTE
1954	010534	010566	000002			MOV	TTY,2(6)	;RESTORE ADDRESS
1955	010540	012605			3\$:	MOV	(6)+,TTY	;RESTORE TTY
1956	010542	016704	000006			MOV	IOTS,R4	;RESTORE R4
1957	010546	000002				RTI		;RETURN

; STYPE MESSAGE TYPEOUT ROUTINE
 ; THIS ROUTINE IS USE TO TYPE ASCII MESSAGES ON THE TTY. THE
 ; CALL CAN BE IN ONE OF 3 FORMS: 1) "TYPE ADR" - TYPES THE
 ; MESSAGE STARTING IN LOCATION "ADR:" 2) "TYPE CHAR" - TYPES
 ; THE ASCII "CHAR", AND 3) "PRINT <<15><12>"MESSAGE"> - TYPES
 ; THE MESSAGE WHICH IS INLINE ASCII.

M04

MAINDEC-11-DZDAA-A
DZDAAA.P11

BJS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 52
TYPE ROUTINE

1958 010550 001000
1959 010552 000000
1960 010554 000000

FILCHR: 1000
.TYPE: 0
IOTIS: 0

;FILCHR=0 (CHAR) FILCHR+1=2 (COUNT)
;CHARACTER TYPE LOCATION

NO4

MAINDEC-11-DZDAA-A
DZDAAA.P11

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 53
TYPE ROUTINE

```

1961 010556 012767 000300 170214 GO.HST: MOV      #300,ICNT      ;HOST COMPUTER ENTRY
1962 010564 012706 000500          MOV      #500,SP      ;INITIALIZE STACK
1963
1964          ;*****
1965          ;TEST 301          POWER FAIL TEST - PF - GOES DOWN - BIT13 SET
1966          ;*****
1967 010570 104400          TST301: SCOPE
1968
1969 010572 012737 010630 000024          MOV      #1$,@#24      ;SET FOR POWER FAIL
1970 010600 012737 000340 000026          MOV      #340,@#36     ;LOCK IT UP
1971 010606 012777 010672 004230          MOV      #2$,@WVECA    ;SET FOR WINDOW TRAP
1972 010614 012777 000502 004202          MOV      #502,@WCSRA   ;OPEN WINDOW AND I/E
1973 010622 000001          WAIT          ;WAIT FOR INTERRUPT
1974 010624 000000          HALT         ;RETURN FROM TRAP
1975 010626 000776          BR          .-2        ;LOCK IT UP
1976 010630 022626          1$: CMP      (6)+,(6)+  ;CLEAR THE STACK FROM THE PF TRAP
1977 010632 012777 010652 004352          MOV      #3$,@PUVECS   ;RESET UP VECTOR
1978 010640 017746 004160          MOV      @WCSRA,-(6)   ;SAVE WCSRA ON THE STACK
1979 010644 010667 004340          MOV      SP,SAVE6     ;SAVE THE SP
1980 010650 000000          HALT         ;WAIT FOR POWER FAIL
1981 010652 016706 004332 3$: MOV      SAVE6,SP     ;RESTORE THE SP
1982 010656 012600          MOV      (6)+,R0      ;GET FOR TYPING
1983 010660 022700 120706          CMP      #120706,R0   ;CHECK THE WCSRA
1984 010664 001401          BEQ      .+4          ;SKIP IF OK
1985 010666 104000          HLT         ;WCSRA NOT 120706 AFTER PF
1986 010670 000404          BR          4$        ;SKIP WINDOW AREA
1987 010672 022626          2$: CMP      (6)+,(6)+  ;CLEAR STACK
1988 010674 017700 004124          MOV      @WCSRA,R0   ;GET FOR TYPING
1989 010700 104000          HLT         ;SHOULD NOT HAVE TRAPPED
1990 010702 012737 000026 000024 4$: MOV      #26,@#24     ;RESET PF VECTOR
1991 010710 005037 177776          CLR      @#PS        ;LOWER PROCESSOR STATUS

```

1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020

010714 104400
010716 012737
010724 012737
010732 012777
010740 012777
010746 000001
010750 000000
010752 000776
010754 022626
010756 012777
010764 017746
010770 010667
010774 000000
010776 016706
011002 012600
011004 104000
011006 000407
011010 022626
011012 017700
011016 022700
011022 001401
011024 104000
011026 012737
011034 000004
011056 000000

010754 000024
000340 000026
011010 004104
000502 004056
010776 004226
004206
004006
120706
000026 000024
011040

15:
35:
25:
45:

MOV #15,2#24
MOV #340,2#26
MOV #25,2#VECA
MOV #502,2#CSRA
WAIT
HALT
BR -2
CMP (6)+,(6)+
MOV #35,2#PUVECS
MOV 2#CSRA,-(6)
MOV SF,SAVE6
HALT
MOV SAVE6,SP
MOV (6)+,R0
HLT
BR 45
CMP (6)+,(6)+
MOV 2#CSRA,R0
CMP #120706,R0
BEQ .+4
HLT
MOV #26,2#24
TYPE ..+2
HALT

:TEST 302 POWER FAIL TEST - NO PF - STAYS UP - BIT13 SET

TST302: SCOPE
:SET FOR POWER FAIL
:LOCK IT UP
:SET FOR WINDOW TRAP
:OPEN WINDOW AND I/E
:WAIT FOR INTERUPT
:RETURN FROM TRAP
:LOCK IT UP
:CLEAR THE STACK FROM THE PF TRAP
:RESET UP VECTOR
:SAVE WCSRA ON THE STACK
:SAVE THE SP
:WAIT FOR POWER FAIL
:RESTORE THE SP
:GET FOR TYPING
:DID NOT POWER FAIL
:SKIP WINDOW AREA
:CLEAR STACK
:GET 2#CSRA FOR TYPING
:IS 2#CSRA = 120706?
:SKIP IF OK
:2#CSRA NOT 120706
:RESET PF VECTOR
:ASCIZ <15><12>"END OF TEST"

```

2021 011060 012767 000310 167712 GO.OTH: MOV #310,ICNT ;OTHER COMPUTER ENTRY
2022
2023 ;*****
2024 ;TEST 311 POWER FAIL TEST - PF - GOES DOWN - BIT13 SET
2025 ;*****
2026 011066 104400 †TST311: SCOPE
2027
2028 011070 012737 011126 000024 MOV #15,2#24 ;SET FOR POWER FAIL
2029 011076 012737 000340 000026 MOV #340,2#26 ;LOCK IT UP
2030 011104 012777 011170 003732 MOV #25,2#WECA ;SET FOR WINDOW TRAP
2031 011112 012777 000502 003704 MOV #502,2#WCSRA ;OPEN WINDOW AND I/E
2032 011120 000001 WAIT ;WAIT FOR INTERRUPT
2033 011122 000000 HALT ;RETURN FROM TRAP
2034 011124 000776 BR ;LOCK IT UP
2035 011126 022626 1S: CMP (6)+,(6)+ ;CLEAR THE STACK FROM THE PF TRAP
2036 011130 012777 011150 004054 MOV #35,2#PUVECS ;RESET UP VECTOR
2037 011136 017746 003652 MOV 2#WCSRA,-(6) ;SAVE WCSRA ON THE STACK
2038 011142 010667 004042 MOV SP,SAVE6 ;SAVE THE SP
2039 011146 000000 HALT ;WAIT FOR POWER FAIL
2040 011150 016706 004034 3S: MOV SAVE6,SP ;RESTORE THE SP
2041 011154 012600 MOV (6)+,R0 ;GET FOR TYPING
2042 011156 022700 120706 CMP #120706,R0 ;CHECK THE WCSRA
2043 011162 001401 BEQ .+4 ;SKIP IF OK
2044 011164 104000 HLT ;WCSRA NOT 120706 AFTER PF
2045 011166 000404 BR 4S ;SKIP WINDOW AREA
2046 011170 022626 2S: CMP (6)+,(6)+ ;CLEAR STACK
2047 011172 017700 003626 MOV 2#WCSRA,R0 ;GET FOR TYPING
2048 011176 104000 HLT ;SHOULD NOT HAVE TRAPPED
2049 011200 012737 000026 000024 4S: MOV #26,2#24 ;RESET PF VECTOR

```

```

2050                                     :*****
2051                                     :TEST 312                               POWER FAIL TEST - PF - GOES DOWN - BIT13 NOT SET
2052                                     :*****
2053 011206 104400                       †TST312: SCOPE
2054
2055 011210 012737 011246 000024          MOV      #15, @#24          ;SET FOR POWER FAIL
2056 011216 012737 000340 000026          MOV      #340, @#26       ;LOCK IT UP
2057 011224 012777 011310 003612          MOV      #25, @WVECA      ;SET FOR WINDOW TRAP
2058 011232 012777 000502 003564          MOV      #502, @WCSRA     ;OPEN WINDOW AND I/E
2059 011240 000001                       WAIT                       ;WAIT FOR INTERRUPT
2060 011242 000000                       HALT                       ;RETURN FROM TRAP
2061 011244 000776                       BR      .-2               ;LOCK IT UP
2062 011246 022626                       1$:  CMP      (6)+, (6)+   ;CLEAR THE STACK FROM THE PF TRAP
2063 011250 012777 011270 003734          MOV      #35, @PUVECS     ;RESET UP VECTOR
2064 011256 017746 003542                 MOV      @WCSRA, -(6)     ;SAVE WCSRA ON THE STACK
2065 011262 010667 003722                 MOV      SP, SAVE6        ;SAVE THE SP
2066 011266 000000                       HALT                       ;WAIT FOR POWER FAIL
2067 011270 016706 003714 3$:           MOV      SAVE6, SP        ;RESTORE THE SP
2068 011274 012600                       MOV      (6)+, R0         ;GET FOR TYPING
2069 011276 022700 000706                 CMP      #706, R0         ;CHECK THE WCSRA
2070 011302 001401                       BEQ     .+4               ;SKIP IF OK
2071 011304 104000                       HLT                       ;WCSRA NOT 706 AFTER PF
2072 011306 000404                       BR      4$                ;SKIP WINDOW AREA
2073 011310 022626                       2$:  CMP      (6)+, (6)+   ;CLEAR STACK
2074 011312 017700 003506                 MOV      @WCSRA, R0       ;GET FOR TYPING
2075 011316 104000                       HLT                       ;SHOULD NOT HAVE TRAPPED
2076 011320 012737 000026 000024 4$:     MOV      #26, @#24        ;RESET PF VECTOR
2077 011326 000004 011332                 TYPE     .,+2            ;.ASCIZ <15><12>"END OF TEST"
2078 011350 000000                       HALT

```



```

2137
2138
2139
2140 011632 104400
2141
2142 011634 012737 011672 000024      MOV      #1$, @#24      ;SET FOR POWER FAIL
2143 011642 012737 000340 000026      MOV      #340, @#26   ;LOCK IT UP
2144 011650 012777 011734 003166      MOV      #2$, @WVECA  ;SET FOR WINDOW TRAP
2145 011656 012777 000502 003140      MOV      #502, @WCSRA ;OPEN WINDOW AND I/E
2146 011664 000001      WAIT     ;WAIT FOR INTERRUPT
2147 011666 000000      HALT    ;RETURN FROM TRAP
2148 011670 000776      BR      .-2          ;LOCK IT UP
2149 011672 022626      1$:    CMP      (6)+, (6)+  ;CLEAR THE STACK FROM THE PF TRAP
2150 011674 012777 011714 003310      MOV      #3$, @PUVECS ;RESET UP VECTOR
2151 011702 017746 003116      MOV      @WCSRA, -(6) ;SAVE WCSRA ON THE STACK
2152 011706 010667 003276      MOV      SP, SAVE6   ;SAVE THE SP
2153 011712 000000      HALT    ;WAIT FOR POWER FAIL
2154 011714 016706 003270      3$:    MOV      SAVE6, SP  ;RESTORE THE SP
2155 011720 012600      MOV      (6)+, R0    ;GET FOR TYPING
2156 011722 022700 120706      CMP      #120706, R0 ;CHECK THE WCSRA
2157 011726 001401      BEQ     .+4          ;SKIP IF OK
2158 011730 104000      HLT     ;WCSRA NOT 120706 AFTER PF
2159 011732 000404      BR      4$          ;SKIP WINDOW AREA
2160 011734 022626      2$:    CMP      (6)+, (6)+  ;CLEAR STACK
2161 011736 017700 003062      MOV      @WCSRA, R0  ;GET FOR TYPING
2162 011742 104000      HLT     ;SHOULD NOT HAVE TRAPPED
2163 011744 012737 000026 000024 4$:    MOV      #26, @#24   ;RESET PF VECTOR
2164
2165 011752 000004 011756      TYPE    .,+2        ;.ASCIZ <15><12>"END OF TEST"
2166 011774 000000      HALT

```

H05

MAINDEC-11-DZDAA-A BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 60
 DZDAAA.P11 TST323 POWER FAIL TEST - PF - GOES DOWN - BIT13 SET

```

2167 011776 012767 000330 166774 GO.NE2: MOV      #330,ICNT      ;NEITHER COMPUTER ENTRY - SECOND
2168
2169                                     ;*****
2170                                     ;TEST 331                    POWER FAIL TEST - NO PF - STAYS UP - BIT13 SET
2171                                     ;*****
2172 012004 104400                    TST331: SCOPE
2173
2174 012006 012737 012044 000024            MOV      #1$,@#24            ;SET FOR POWER FAIL
2175 012014 012737 000340 000026            MOV      #340,@#26           ;LOCK IT UP
2176 012022 012777 012100 003014            MOV      #2$,@WVECA           ;SET FOR WINDOW TRAP
2177 012030 012777 000502 002766            MOV      #502,@WCSRA        ;OPEN WINDOW AND I/E
2178 012036 000001                    WAIT                        ;WAIT FOR INTERRUPT
2179 012040 000000                    HALT                        ;RETURN FROM TRAP
2180 012042 000776                    BR                         ;LOCK IT UP
2181 012044 022626                    1$: CMP      (6)+,(6)+        ;CLEAR THE STACK FROM THE PF TRAP
2182 012046 012777 012066 003136            MOV      #3$,@PUVECS        ;RESET UP VECTOR
2183 012054 017746 002744            MOV      @WCSRA,-(6)        ;SAVE WCSRA ON THE STACK
2184 012060 010667 003124            MOV      SP,SAVE6           ;SAVE THE SP
2185 012064 000000                    HALT                        ;WAIT FOR POWER FAIL
2186 012066 016706 003116            3$: MOV      SAVE6,SP        ;RESTORE THE SP
2187 012072 012600                    MOV      (6)+,R0            ;GET FOR TYPING
2188 012074 104000                    HLT                        ;DID NOT POWER FAIL
2189 012076 000407                    BR                         ;SKIP WINDOW AREA
2190 012100 022626                    2$: CMP      (6)+,(6)+        ;CLEAR STACK
2191 012102 017700 002716            MOV      @WCSRA,R0        ;GET @WCSRA FOR TYPING
2192 012106 022700 120706            CMP      #120706,R0        ;IS @WCSRA = 120706?
2193 012112 001401                    BEQ                        ;SKIP IF OK
2194 012114 104000                    HLT                        ;@WCSRA NOT 120706
2195 012116 012737 000026 000024 4$: MOV      #26,@#24            ;RESET PF VECTOR
  
```

```

2196                                     ;*****
2197                                     ;TEST 332
2198                                     ;*****
2199 012124 104400                       †TST332: SCOPE
2200
2201 012126 012737 012164 000024         MOV    #15, @#24           ;SET FOR POWER FAIL
2202 012134 012737 000340 000026         MOV    #340, @#26        ;LOCK IT UP
2203 012142 012777 012226 002674         MOV    #25, @WVECA      ;SET FOR WINDOW TRAP
2204 012150 012777 000502 002646         MOV    #502, @WCSRA     ;OPEN WINDOW AND I/E
2205 012156 000001                       WAIT                   ;WAIT FOR INTERRUPT
2206 012160 000000                       HALT                   ;RETURN FROM TRAP
2207 012162 000776                       BR                      ;LOCK IT UP
2208 012164 022626                       1$: CMP    (6)+, (6)+     ;CLEAR THE STACK FROM THE PF TRAP
2209 012166 012777 012206 003016         MOV    #35, @PUVECS     ;RESET UP VECTOR
2210 012174 017746 002624                 MOV    @WCSRA, -(6)     ;SAVE WCSRA ON THE STACK
2211 012200 010667 003004                 MOV    SP, SAVE6        ;SAVE THE SP
2212 012204 000000                       HALT                   ;WAIT FOR POWER FAIL
2213 012206 016706 002776                       3$: MOV    SAVE6, SP     ;RESTORE THE SP
2214 012212 012600                       MOV    (6)+, R0         ;GET FOR TYPING
2215 012214 022700 000706                 CMP    #706, R0         ;CHECK THE WCSRA
2216 012220 001401                       BEQ    .+4              ;SKIP IF OK
2217 012222 104000                       HLT                    ;WCSRA NOT 706 AFTER PF
2218 012224 000404                       BR     4$              ;SKIP WINDOW AREA
2219 012226 022626                       2$: CMP    (6)+, (6)+     ;CLEAR STACK
2220 012230 017700 002570                 MOV    @WCSRA, R0       ;GET FOR TYPING
2221 012234 104000                       HLT                    ;SHOULD NOT HAVE TRAPPED
2222 012236 012737 000026 000024 4$: MOV    #26, @#24        ;RESET PF VECTOR

```

```

2223                                     ;*****
2224                                     ;TEST 333          POWER FAIL TEST - PF - GOES DOWN - BIT13 SET
2225                                     ;*****
2226 012244 104400                       †TST333: SCOPE
2227
2228 012246 012737 012304 000024         MOV    #1$, @#24           ;SET FOR POWER FAIL
2229 012254 012737 000340 000026         MOV    #340, @#26        ;LOCK IT UP
2230 012262 012777 012346 002554         MOV    #2$, @WVECA       ;SET FOR WINDOW TRAP
2231 012270 012777 000502 002526         MOV    #502, @WCSRA     ;OPEN WINDOW AND I/E
2232 012276 000001                       WAIT                   ;WAIT FOR INTERRUPT
2233 012300 000000                       HALT                   ;RETURN FROM TRAP
2234 012302 000776                       BR                      ;LOCK IT UP
2235 012304 022626                       1$: CMP    (6)+, (6)+     ;CLEAR THE STACK FROM THE PF TRAP
2236 012306 012777 012326 002676         MOV    #3$, @PUVECS     ;RESET UP VECTOR
2237 012314 017746 002504                 MOV    @WCSRA, -(6)     ;SAVE WCSRA ON THE STACK
2238 012320 010667 002664                 MOV    SP, SAVE6       ;SAVE THE SP
2239 012324 000000                       HALT                   ;WAIT FOR POWER FAIL
2240 012326 016706 002656                       3$: MOV    SAVE6, SP     ;RESTORE THE SP
2241 012332 012600                       MOV    (6)+, R0        ;GET FOR TYPING
2242 012334 022700 120706                 CMP    #120706, R0     ;CHECK THE WCSRA
2243 012340 001401                       BEQ    .+4             ;SKIP IF OK
2244 012342 104000                       HLT                      ;WCSRA NOT 120706 AFTER PF
2245 012344 000404                       BR    4$               ;SKIP WINDOW AREA
2246 012346 022626                       2$: CMP    (6)+, (6)+     ;CLEAR STACK
2247 012350 017700 002450                 MOV    @WCSRA, R0     ;GET FOR TYPING
2248 012354 104000                       HLT                      ;SHOULD NOT HAVE TRAPPED
2249 012356 012737 000026 000024         4$: MOV    #26, @#24    ;RESET PF VECTOR
2250
2251 012364 000004 012370                 TYPE    ,.+2           ;.ASCIZ <15><12>"END OF TEST"
2252 012406 000000                       HALT

```

K05

MAINDEC-11-DZDAA-A
DZDAAA.P11

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 63
SCOPE LOOP HANDLER

```

2253           :          $SCOPE          SCOPE LOOP HANDLER
2254
2255           ;THIS ROUTINE HANDLES THE ITERATIONS, LOOPING, ERROR
2256           ;LOOPING, AND THE DISPLAYING OF THE TEST NUMBER.
2257
2258           ;"SCOPE" IS PLACED BETWEEN EACH SUBTEST IN THE TEST AND
2259           ;RECORDS THE STARTING ADDRESS OF THE SUBTEST IN "LAD:"
2260
2261 012410 032737 040000 177570 TRAPS: BIT      #SW14, @#SWR      ;LOOP ON TEST?
2262 012416 001047          BNE      KITS$          ;LOOP ON TEST IS SET
2263 012420 000416          BR       3$          ;SKIP - NOP FOR XOR TESTER
2264 012422 013746 000004          MOV     @#4, -(6)      ;PUSH @#4 ON STACK
2265 012426 012737 012446 000004          MOV     #4$, @#4      ;SET FOR TIMEOUT
2266 012434 005737 177060          TST     @#177060     ;ERROR ON XOR?
2267 012440 012637 000004          MOV     (6)+, @#4     ;POP STACK INTO @#4
2268 012444 000422          BR       SVLAD$      ;NO ERROR - GO TO NEXT TEST
2269 012446 022626          4$:    CMP     (6)+, (6)+      ;CLEAR STACK
2270 012450 012637 000004          MOV     (6)+, @#4     ;POP STACK INTO @#4
2271 012454 000430          BR       KITS$          ;ERROR - LOOP ON TEST
2272 012456 032737 004000 177570 3$:    BIT     #SW11, @#SWR     ;KILL ITERATIONS
2273 012464 001012          BNE      SVLAD$      ;YES - KILL ITERATIONS
2274 012466 105767 166307          TSTB   ICNT+1        ;FIRST ONE?
2275 012472 001404          BEQ     2$          ;BRANCH IF FIRST
2276 012474 126767 000126 166277          CMPB   TIMES, ICNT+1 ;DONE?
2277 012502 001015          BNE      KITS$          ;BRANCH IF NOT
2278 012504 112767 000001 166267 2$:    MCVB   #1, ICNT+1     ;FIRST ITERATION
2279 012512 105267 166262          SVLAD$: INCB     ICNT      ;COUNT TEST NUMBERS
2280 012516 011667 166266          MOV     (6) LAD      ;SAVE LOOP ADDRESS
2281 012522 016737 166252 177570          MOV     ICNT, @#DISPLAY ;DISPLAY TEST NO. AND ITERATION COUNT
2282 012530 004767 000034          JSR     PC, $TRACE
2283 012534 000002          RTI
2284
2285 012536 105267 166237          KITS$:  INCB     ICNT+1      ;INC THE ITERATION COUNT
2286 012542 016737 166232 177570 OVER$: MOV     ICNT, @#DISPLAY ;SET UP DISPLAY
2287 012550 004767 000014          JSR     PC, $TRACE
2288 012554 005767 166230          TST     LAD          ;FIRST ONE?
2289 012560 001754          BEQ     SVLAD$      ;YES
2290 012562 016716 166222          MOV     LAD, (6)     ;FLDGE RETURN ADDRESS
2291 012566 000002          RTI
2292
2293 012570 032737 000400 177570 $TRACE: BIT     #SW8, @#SWR      ;CHECK BIT 8
2294 012576 001001          BNE     .+4          ;SKIP IF TRACE
2295 012600 000207          RTS     PC          ;RETURN
2296 012602 000004 012606          TYPE   .+2          ;.ASCIZ <15><12>"+ "
2297 012612 116705 166162          MOVB   ICNT, TTY     ;GET BYTE FOR TYPING
2298 012616 042705 177400          BIC     #177400, TTY ;CLEAR JUNK
2299 012622 000167 001760          JMP     PRINTS      ;TYPE IT
2300
2301 012626 000001          TIMES: 1          ;RUN 1 TIMES

```

2302	012630	012737	000340	000006	SETUP:	MOV	#340, @#6	:LOCK UP 4
2303	012636	012737	010402	000020		MOV	#IOT\$, @#20	:SET UP IOT VECTOR
2304	012644	012737	000340	000022		MOV	#340, @#22	:LOCK UP
2305	012652	012737	012410	000034		MOV	#TRAP\$, @#34	:SET TRAP VECTOR
2306	012660	012737	000340	000036		MOV	#340, @#36	:LOCK UP
2307	012666	012737	014416	000030		MOV	#EMT\$, @#30	:SET EMT VECTOR
2308	012674	012737	000340	000032		MOV	#340, @#32	:LOCK UP
2309	012702	005267	002300			INC	ONCE	:CHECK ONCE ONLY SWITCH
2310	012706	001402				BEQ	.+6	:SKIP JMP
2311	012710	000167	001364			JMP	SET1	:SKIP IF NOT FIRST TIME
2312	012714	000004	012720			TYPE	.+2	:.ASCIZ <15><12><12>"MAINDEC-11-DZDAA-A"<15><12><12>
2313	012752	012737	012766	000004	1\$:	MOV	#7\$, @#4	:SET FOR TRAPPING
2314	012760	005777	002040			TST	@WCSRA	:IS THE ADDRESS LEGAL
2315	012764	000440				BR	SET2	:YES
2316	012766	022626			7\$:	CMP	(6)+, (6)+	:CLEAR STACK
2317	012770	012767	164000	002026		MOV	#164000, WCSRA	:SET WCSRA FOR SCANNING
2318	012776	012701	000004			MOV	#4, R1	:SET COUNT
2319	013002	012737	013016	000004		MOV	#5\$, @#4	:SET FOR TIMEOUT
2320	013010	005777	002010		6\$:	TST	@WCSRA	:IS IT THERE?
2321	013014	000424				BR	SET2	:YES - GET OUT
2322	013016	022626			5\$:	CMP	(6)+, (6)+	:CLEAR STACK
2323	013020	062767	000020	001776		ADD	#20, WCSRA	:BUMP ADDRESS
2324	013026	005301				DEC	R1	:DEC COUNT
2325	013030	001367				BNE	6\$:LOOP UNTIL FOUND
2326	013032	000004	013036			TYPE	.+2	:.ASCIZ "WCSRA NOT FOUND"<15><12><12>
2327	013062	000000				HALT		:NO WCSRA
2328	013064	000732				BR	1\$:TRY IT AGAIN
2329	013066				SET2:			
2330	013066	000004	013072			TYPE	.+2	:.ASCIZ "WCSRA = "
2331	013104	016705	001714			MOV	WCSRA, TTY	:TYPE WCSRA IN OCTAL
2332	013110	004767	001462			JSR	PC, PRINTR	:TYPE LEADING ZERO'S
2333	013114	000004	013120			TYPE	.+2	:.ASCIZ " "
2334	013124	000402				BR	32\$:SKIP
2335	013126	000000			31\$:	HALT		:WINDOW ADDRESS TIMED OUT
2336	013130	022626				CMP	(6)+, (6)+	:CLEAR STACK
2337	013132	012737	013126	000004	32\$:	MOV	#31\$, @#4	:SET 4 FOR TIMEOUT
2338	013140	016700	001660			MOV	WCSRA, RO	:GET THE FIRST ADD
2339	013144	012701	015026			MOV	#WDBRA, R1	:GET TABLE ADDRESS
2340	013150	005720			60\$:	TST	(0)+	:ADD 2
2341	013152	010021				MOV	RO, (1)+	:GET THE DATA
2342	013154	022701	015044			CMP	#WVECA, R1	:END?
2343	013160	001373				BNE	60\$:LOOP UNTIL DONE
2344	013162	012737	013172	000004		MOV	#20\$, @#4	:SET TIMEOUT RETURN
2345	013170	000402				BR	21\$:SKIP IT
2346	013172	000000			20\$:	HALT		:WINDOW IS NOT AT THIS ADDRESS
2347	013174	022626				CMP	(6)+, (6)+	:CLEAR THE STACK
2348	013176	012777	177777	001630	21\$:	MOV	#-1, @WRARA	:FIND THE WINDOW SIZE
2349	013204	017700	001624			MOV	@WRARA, RO	:SAVE THE SIZE
2350	013210	010067	001664			MOV	RO, SIZE	:SAVE THE SIZE FOR POSTERITY
2351	013214	005400				NEG	RO	:BUMP IT TO 1 BIT ONLY
2352	013216	000300				SWAB	RO	:GET UPPER BITS
2353	013220	006200				ASR	RO	:MOVE IT INTO POSITION
2354	013222	005001				CLR	R1	:INIT FOR OFFSET
2355	013224	005700			54\$:	TST	RO	:IS IT 0
2356	013226	001404				BEQ	43\$:YES - GET OUT
2357	013230	006200				ASR	RO	:COUNT BITS

M05

MAINDEC-11-DZDAA-A
DZDAAA.P11

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 65
SUBROUTINES

2358	013232	062701	000002			AUD	#2,R1	;KEEP COUNT
2359	013236	000772				BR	54\$;LOOP
2360	013240	016167	015106	001734	43\$:	MOV	WIND(1),SIZ	;GET THE SIZE
2361	013246	010167	001726			MOV	R1,OFFSET	;SAVE OFFSET
2362	013252	000004	015202			TYPE	,SIZ	;TYPE THE WINDOW SIZE
2363	013256	000004	013262			TYPE	.+2	;ASCIZ "K WINDOW AT "
2364	013300	017700	001532			MOV	QWADRAX,RO	;GET WINDOW ADDRESS
2365	013304	005067	001666			CLR	MM	;CLEAR MEMORY MANAGEMENT FLAG
2366	013310	005067	001506			CLR	MX	;GET READY TO TYPE
2367	013314	012701	000005			MOV	#5,R1	;INIT COUNT
2368	013320	006300			SET3:	ASL	RO	;MOVE IT LEFT
2369	013322	006167	001474			ROL	MX	;PICK UP DROPPED BIT
2370	013326	005301				DEC	R1	;COUNT IT
2371	013330	001373				BNE	SET3	;LOOP UNTIL 0
2372	013332	017700	001500			MOV	QWADRAX,RO	;GET AGAIN
2373	013336	006300				ASL	RO	;GET FOR
2374	013340	006300				ASL	RO	;TYPING
2375	013342	010067	001502			MOV	RO,WADRA	;SAVE THE WINDOW ADDRESS
2376	013346	010005				MOV	RO,TTY	;TYPE RO WITH MX AS 18 BIT ADDRESS
2377	013350	004767	001256			JSR	PC,PRINTA	;GO TO ADDRESS PRINTER
2378	013354	012737	013550	000004		MOV	#34\$,Q#4	;SET TIME OUT TRAP
2379	013362	005737	177572			TST	Q#SR0	;CHECK FOR MM
2380	013366	005067	156746			CLR	KIPAR0	;MAP TO 0
2381	013372	012767	077406	156700		MOV	#77406,KIPDR0	;READ/WRITE 4K
2382	013400	012702	000005			MOV	#5,R2	;SET THE COUNT
2383	013404	012701	172304			MOV	#KIPDR2,R1	;GET THE ADDRESS
2384	013410	012721	077406		47\$:	MOV	#77406,(1)+	;SET THE DPR
2385	013414	005302				DEC	R2	;CHECK COUNT
2386	013416	001374				BNE	47\$;LOOP
2387	013420	012767	007600	156730		MOV	#7600,KIPAR7	;MAP TO BANK 37
2388	013426	012767	077406	156662		MOV	#77406,KIPDR7	;READ/WRITE 4K
2389	013434	005267	001444			INC	FLAG	;SET THE FLAG
2390	013440	017700	001372			MOV	QWADRAX,RO	;GET ADDRESS
2391	013444	006000				ROR	RO	;MAKE
2392	013446	006000				ROR	RO	;IT
2393	013450	006000				ROR	RO	;PAGE
2394	013452	006000				ROR	RO	;ADDRESS
2395	013454	042700	170000			BIC	#170000,RO	;CLEAR JUNK
2396	013460	010067	156660			MOV	RO,KIPAR2	;SET IT
2397	013464	012702	000004			MOV	#4,R2	;SET THE COUNT
2398	013470	012701	172346			MOV	#KIPAR3,R1	;GET ADDRESS
2399	013474	062700	000200		57\$:	ADD	#200,RO	;GO TO NEXT
2400	013500	010021				MOV	RO,(1)+	;SET IN THE ADDRESS
2401	013502	005302				DEC	R2	;COUNT
2402	013504	001373				BNE	57\$;LOOP
2403	013506	012767	040000	001334		MOV	#40000,WADRA	;FUDGE ADDRESS
2404	013514	016767	156564	156560		MOV	KIPDR2,KIPDR1	
2405	013522	016701	156616			MOV	KIPAR2,R1	
2406	013526	016702	001446			MOV	OFFSET,R2	
2407	013532	066201	015160			ADD	ADR3(2),R1	
2408	013536	010167	156600			MOV	R1,KIPAR1	
2409	013542	010667	001430			MOV	SP,MM	
2410	013546	000401				BR	50\$	
2411	013550	022626			34\$:	CMP	(6)+,(6)+	;RESTORE STACK
2412	013552	012737	000006	000004	50\$:	MOV	#6,Q#4	;RESET 4
2413	013560	017700	001254			MOV	QWVARA,RO	;GET VECTOR ADDRESS

N05

MAINDEC-11-DZDAA-A
DZDAAA.P11

BUS WINDOW STATIC TEST MACY11 27(732) 05-OCT-76 10:45 PAGE 66
SUBROUTINES

2414	013564	012737	000006	000004		MOV	#6, @#4	; RESET 4
2415	013572	000004	013576			TYPE	.+2	; .ASCIZ " VECTOR = "
2416	013614	022700	001000			CMP	#1000, R0	; CHECK FOR UNDER 1000
2417	013620	002403				BLT	25\$; GO TO ERROR ROUTINE
2418	013622	022700	000136			CMP	#136, R0	; CHECK FOR OVER 136
2419	013626	002403				BLT	SET4	; SKIP IF IN RANGE
2420	013630	000000			25\$:	HALT		; VECTOR IS NOT BETWEEN 140-776
2421	013632	000167	000442			JMP	SET1	; GET OUT IF CONTINUED
2422	013636	010067	001202		SET4:	MOV	R0, WVECA	; SAVE THE VECTOR ADDRESS
2423	013642	005720				TST	(0)+	; ADD 2
2424	013644	010067	001176			MOV	R0, WVECA+2	; SET UP STATUS
2425	013650	017705	001164			MOV	@WVARA, TTY	; TYPE @WVARA IN OCTAL
2426	013654	004767	000726			JSR	PC, PRINTS	; AND SUPPRESS LEADING ZERO'S
2427	013660	000004	013664			TYPE	.+2	; .ASCIZ <15><12>
2428	013670	022716	001024			CMP	#ONEB1, (6)	; ONLY 1 ADDRESS NEEDED?
2429	013674	001402				BEQ	30\$; NO - CONTINUE
2430	013676	000167	000364			JMP	SET5	
2431	013702	016700	001116		30\$:	MOV	WCSRA, R0	; GET STARTING ADDR OF "A" REGISTERS
2432	013706	062700	000020			ADD	#20, R0	; DETERMINE START OF "B" REGISTERS
2433	013712	042700	000100			BIC	#100, R0	; CLEAR UNWANTED CARRY
2434	013716	010067	001130			MOV	R0, WCSR8	; SAVE IT
2435	013722	000004	013726			TYPE	.+2	; .ASCIZ /WCSR8 = /
2436	013740	016705	001106			MOV	WCSR8, TTY	; TYPE WCSR8 IN OCTAL
2437	013744	004767	000626			JSR	PC, PRINTR	; TYPE LEADING ZERO'S
2438	013750	000004	013754			TYPE	.+2	; .ASCIZ / /
2439	013760	000402				BR	10\$; SKIP IT
2440	013762	000000			11\$:	HALT		; "B" WINDOW ADDRESS TIMED OUT
2441	013764	022626				CMP	(6)+, (6)+	; RESTORE STACK
2442	013766	012737	013762	000004	10\$:	MOV	#11\$, @#4	; SET 4 FOR TIMEOUT
2443	013774	016700	001052			MOV	WCSR8, R0	; GET THE FIRST ADDR
2444	014000	012701	015054			MOV	#WDBRBX, R1	; GET TABLE ADDR
2445	014004	005720			12\$:	TST	(0)+	; ADD 2
2446	014006	010021				MOV	R0, (1)+	; LOAD "B" ADDRESS
2447	014010	022701	015072			CMP	#WVECB, R1	; END?
2448	014014	001373				BNE	12\$; LOOP UNTIL DONE
2449	014016	012737	014026	000004		MOV	#13\$, @#4	; SET TIMEOUT RETURN
2450	014024	000402				BR	14\$; SKIP IT
2451	014026	000000			13\$:	HALT		; WINDOW IS NOT AT THIS ADDRESS
2452	014030	022626				CMP	(6)+, (6)+	; RESTORE STACK
2453	014032	012777	177777	001022	14\$:	MOV	#-1, @WRARB	; FIND THE WINDOW SIZE
2454	014040	017700	001016			MOV	@WRARB, R0	; SAVE THE SIZE
2455	014044	005400				NEG	R0	; BUMP IT TO ONE BIT ONLY
2456	014046	000300				SWAB	R0	; GET UPPER BITS
2457	014050	006200				ASR	R0	; MOVE IT INTO POSITION
2458	014052	005001				CLR	R1	; INIT FOR OFFSET
2459	014054	005700			SET6:	TST	R0	; IS TI 0?
2460	014056	001404				BEQ	16\$; YES - GET OUT
2461	014060	006200				ASR	R0	; COUNT BITS
2462	014062	062701	000002			ADD	#2, R1	; KEEP COUNT
2463	014066	000772				BR	SET6	
2464	014070	016167	015106	00104	16\$:	MOV	WIND(1), SIZ	; GET THE SIZE
2465	014076	000004	015202			TYPE	, SIZ	; TYPE THE WINDOW SIZE
2466	014102	000004	014106			TYPE	.+2	; .ASCIZ /K WINDOW AT /
2467	014124	017700	000734			MOV	@WADRBX, R0	; GET WINDOW ADDR
2468	014130	005067	000666			CLR	MX	; GET READY TO TYPE
2469	014134	012701	000005			MOV	#5, R1	; INIT COUNT

014140	006300			175:	ASL	R0	: MOVE IT LEFT
014142	006167	000654			ROL	MX	: PICK UP DROPPED BIT
014146	005301				DEC	R1	: COUNT IT
014150	001373				BNE	175	: LOOP UNTIL 0
014152	017700	000706			MOV	2WADR BX, R0	: GET ADDRESS AGAIN
014156	006300				ASL	R0	: POSITION FOR
014160	006300				ASL	R0	: TYPING
014162	010005				MOV	R0, TTY	: TYPE R0 WITH MX AS 18 BIT ADDRESS
014164	004767	000442			JSR	PC, PRINTA	: GO TO ADDRESS PRINTER
014170	017700	000672			MOV	2WVARB, R0	: GET THE B VECTOR
014174	000004	014200			TYPE	.+2	: .ASCIZ / VECTOR = /
014216	017705	000644			MOV	2WVARB, TTY	: TYPE 2WVARB IN OCTAL
014222	004767	000360			JSR	PC, PRINTS	: AND SUPPRESS LEADING ZERO'S
014226	000004	014232			TYPE	.+2	: .ASCIZ (15)<(12)
014236	022700	001000			CMP	#1000, R0	: CHECK FOR UNDER 1000
014242	002403				BLT	275	: ERROR IF OVER 1000
014244	022700	000136			CMP	#136, R0	: CHECK FOR UNDER 136
014250	002401				BLT	285	: SKIP IF UNDER 140
014252	000000			275:	HALT		: ADDRESS NOT 140-776
014254	010067	000612		295:	MOV	R0, WVECB	: SAVE THE ADDRESS
014260	005720				TST	(0)+	: ADD 2
014262	010067	000606			MOV	R0, WVECB+2	: SAVE THE STATUS ADDRESS
014266	012737	000006	000004	SET5:	MOV	#6, 2#4	: RESTORE 4
014274	012706	000476			MOV	#476, SP	: RESET SP
014300	005067	164504		SET1:	CLR	LAD	: CLEAR THE LOOP ADDRESS
014304	005067	164470			CLR	ICNT	: CLEAR THE COUNT
014310	016777	000532	000526		MOV	WVECA+2, 2WVECA	: LOCK OUT DALL TRAPS
014316	005077	000524			CLR	2WVECA+2	: HALT ADDRESS
014322	016777	000546	000542		MOV	WVECB+2, 2WVECB	: LOCK OUT OTHER ADDRESS
014330	005077	000540			CLR	2WVECB+2	: HALT ADDRESS
014334	012737	000006	000004		MOV	#6, 2#4	: RESET 4
014342	005767	000536			TST	FLAG	: MM IN USE?
014346	001403				BEQ	35	: NO - GET OUT
014350	012737	000001	177572		MOV	#1, 2#SRC	: TURN IT ON
014356	000207			35:	RTS	PC	: RETURN
014360	012704	000020		TIMO:	MOV	#20, R4	: SET COUNT FOR DOUBLE LOOP
014364	005005			15:	CLR	R5	: ZERO COUNT
014366	032777	010000	000430	25:	BIT	#10000, 2WCSRA	: CHECK FOR NEW DATA B
014374	001401				BEQ	35	: SKIP IF NOT SET
014376	000207			45:	RTS	PC	: RETURN
014400	005205			35:	INC	R5	: BUMP COUNT
014402	001371				BNE	25	: LOOP IF NO TIMEOUT
014404	005304				DEC	R4	: DEC OTHER COUNT
014406	001366				BNE	15	: LOOP IF NO TIMEOUT
014410	011600				MOV	(6), R0	: GET THE ADDRESS
014412	104000				HLT		: NO TIMEOUT
014414	000000				HLT		

2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556

014416 032737 002000 177570
014424 001402
014426 000004 000007
014428 005267 164344
014430 032737 020000 177570
014432 001026
014434 000004 014452
014436 011667 164330
014438 162767 000002 164322
014440 117767 164316 000036
014442 016705 164310
014444 004767 000070
014446 000004 014512
014448 004767 000014
014450 005737 177570
014452 100001
014454 000000
014456 000002
014534 000000
014536
014538 010005
014540 004767 000032
014542 005767 177764
014544 001405
014546 000004 000040
014548 010105
014550 004767 000012
014552 000207

: SHLT ERROR TYPEOUT HANDLER
: THIS ROUTINE PRINTS OUT ERROR MESSAGES STARTING WITH THE
: ADDRESS OF THE "HLT". IT ALSO COUNTS THE NUMBER OF ERRORS
: AND HAS THE CAPABILITY OF LOOPING ON ERROR, BELL ON ERROR,
: "HALT" ON ERROR, AND INHIBIT TYPEOUTS. AN OPTIONAL ARGUMENT
: (HLT+3) WILL BE PLACED IN "HLTCTS:" FOR ADDITIONAL TYPEOUTS.
EMTS: BIT #SW10,2#SWR ;BELL ON ERROR?
BEQ 1\$;NO - SKIP
TYPE BELL ;RING BELL
1\$: INC ERRORS ;COUNT THE NUMBER OF ERRORS
BIT #SW13,2#SWR ;SKIP TYPEOUT IF SET
BNE 2\$;SKIP TYPEOUTS
TYPE .+2 ;ASCIZ <15><12>
MOV (6),HLTADR ;PUT ADDRESS OF INSTRUCTION ON STACK
SUB #2,HLTADR ;FUDGE ADDRESS
MOVB 2#HLTADR,HLTCTS ;GET HLT ARGUMENT
MOV HLTADR,TTY ;TYPE HLTADR IN OCTAL
JSR PC,PRINTR ;TYPE LEADING ZERO'S
TYPE .+2 ;ASCIZ " "
JSR PC,ROUTIN ;GO TO USER ERROR ROUTINE
2\$: TST 2#SWR ;HALT ON ERROR
BPL .+4 ;SKIP IF CONTINUE
HALT ;HALT ON ERROR!
RTI ;RETURN
HLTCTS: 0 ;HLT ARGUMENT
ROUTIN: MOV R0,TTY ;TYPE R0 IN OCTAL
JSR PC,PRINTR ;TYPE LEADING ZERO'S
TST HLTCTS ;CHECK FOR 2 WORD TYPE
BEQ 1\$;RETURN IF 0
TYPE 40 ;TYPE A SPACE
MOV R1,TTY ;TYPE R1 IN OCTAL
JSR PC,PRINTR ;TYPE LEADING ZERO'S
1\$: RTS PC ;RETURN TO SCOPE

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

014566 012767 170101 000202
014574 000411
014576 112767 000001 000172
014604 000402
014606 005067 000164
014612 112767 177772 000157
014620 010446
014622 012704 015000
014626 105014
014630 000432
014632 010446
014634 012704 015000
014640 116714 000156
014644 006305
014646 006305
014650 006305
014652 106214
014654 006005
014656 106214
014660 006005
014662 152724 000060
014666 012767 175401 000102
014674 105014
014676 032767 000100 000072
014704 001004
014706 006105
014710 106114
014712 006105
014714 106114
014716 006105
014720 106114
014722 105714
014724 001402
014726 105267 000044
014732 105767 000040
014736 001402
014740 152724 000060
014744 105267 000027
014750 001351
014752 022704 015000
014756 001002
014760 112724 000060
014764 105014
014766 000004 015000
014772 012604
014774 000207
014776 000012
015022 000000

: SUCTAL OCTAL TYPEOUT ROUTINE

: THIS ROUTINE IS USED TO TYPE AN OCTAL NUMBER ON THE TTY. IT WILL TYPE
: ALL 6 CHARACTERS, SUPPRESS LEADING ZEROES, TYPE AN 18 BIT ADDRESS, OR TYPE
: THE 16 BITS. IT IS CALLED VIA THE DUMP, SDUMP, DUMP18, OR BITYPE MACRO'S.

```

BITYPS: MOV #170101,.PR ;SET BIT FLAG ANS 16. CHARACTER COUNT
        BR .PTIT ;NOW TYPE IT IN BIT FORM
PRINTR: MOVB #1,.PR ;SET ZERO FILL SWITCH
        BR .+6 ;SKIP
PRINTS: CLR .PR ;SUPPRESS LEADING ZERO'S
        MOVB #-6,.PR+1 ;SET COUNT
.PTIT: MOV R4 -(6) ;SAVE R4
        MOV #.PR+2,R4 ;SET POINTER TO FIRST ASCII CHAR.
        CLRB (4) ;CLEAR FIRST BYTE
        BR .PRF ;ROTATE FIRST BIT
PRINTA: MOV R4 -(6) ;SAVE R4
        MOV #.PR+2,R4 ;SET UP POINTER TO OUTPUT AREA
        MOVB MX,(4) ;MX CONTAINS UPPER 5 BITS
        ASL TTY ;GET RID
        ASL TTY ;OF 3
        ASL TTY ;JUNK BITS
        ASRB (4) ;GET BIT13
        ROR TTY ;PACK IT
        ASRB (4) ;GET BIT14
        ROR TTY ;PACK IT
        BISB #'0,(4)+ ;MAKE IT ASCII
        MOV #175401,.PR ;-5, 1 - 5 BYTES AND FILL
.PRL: CLRB (4) ;CLEAR BYTE OF CHARACTER
        BIT #100,.PR ;BIT TYPING MODE?
        BNE .PRF ;YES - SKIP 2 ROTATES
        ROL TTY ;ROTATE BIT INTO C
        ROLB (4) ;PACK IT
        ROL TTY ;ROTATE BIT INTO C
        ROLB (4) ;PACK IT
.PRF: ROL TTY ;ROTATE BIT INTO C
        ROLB (4) ;PACK IT
        TSTB (4) ;IS IT ZERO?
        BEQ .+6 ;SKIP INC
        INCB .PR ;SET FILL SWITCH
        TSTB .PR ;CHECK FILL SWITCH
        BEQ .+6 ;SKIP BITSET
        BISB #'0,(4)+ ;MAKE INTO ASCII CHAR
        INCB .PR+1 ;INC COUNT
        BNE .PRL ;REPEAT
        CMP #.PR+2,R4 ;EMPTY BUFFER?
        BNE .+6 ;SKIP IF NOT
        MOVB #'0,(4)+ ;LOAD 1 ZERO
        CLRB (4) ;NULL TERMINATOR
        TYPE .PR+2 ;TYPE IT
        MOV (6)+,R4 ;RESTORE R4
        RTS PC ;RETURN
.PR: .BLKW 12 ;COUNT, SWITCH, AND OUTPUT BUFFER
MX: 0 ;MEMORY EXTENSION BITS
    
```


2667		017440		.	=	17440		
2668								
2669	017440	000005		OPEN:	RESET			;ZAP THE WORLD
2670	017442	012706	000500		MOV	#500,SP		;INITIALIZE THE STACK
2671	017446	012767	177777		MOV	#-1,ONCE		;RESET
2672	017454	012767	000777		MOV	#777,BR.		;RESET BR
2673	017462	004767	161074		JSR	PC,REGSCN		;GET THE REGISTER
2674	017466	010067	000046		MOV	RO,BR.1		
2675	017472	012710	000402		MOV	#402,(0)		;OPEN THE WINDOW
2676	017476	000777		BR.:	BR	.		;WAIT
2677	017500	000240			NOP			
2678	017502	000240			NOP			
2679	017504	000240			NOP			
2680	017506	000240			NOP			
2681	017510	000240			NOP			
2682	017512	005077	000022		CLR	BR.1		;CLEAR WCSRA
2683	017516	012701	000100		MOV	#100,R1		
2684	017522	005000		20\$:	CLR	RO		;WAIT
2685	017524	005300		10\$:	DEC	RO		;BEFORE
2686	017526	001376			BNE	10\$;STARTING
2687	017530	005301			DEC	R1		
2688	017532	001373			BNE	20\$		
2689	017534	000137	003410		JMP	2\$BEGIN		;SELF START
2690	017540	000000		BR.1:	0			
2691								
2692		000001			.END			

MX = 015022
N = 000334

OFFSET 01520C
ONCE 015206
ONEBUS 001014
ONEB1 001024
OPEN 017440
OVERS 012542
PC =%000007

PCNT 001004
PRINTA 014632
PRINTR 014576
PRINTS 014606
PS = 177776
PLVECS 015212
QG = 000001

R = 000100

REGSCH 000562
ROUTIN 014536
RO =%000000

2366*	2369*	2468*	2471*	2575	2510*									
427*	514	518*	548	552*	582	586*	616	620*	650	654*	684	688*		
721	725*	761	765*	801	805*	874	878*	947	951*	994*	1004	1008*		
1048	1052*	1076	1080*	1120	1124*	1209	1213*	1247	1251*	1281	1285*	1387		
1391*	1425	1429*	1457	1461*	1486	1490*	1508	1512*	1518	1522*	1550*	1552		
1556*	1581	1585*	1609	1613*	1653	1657*	1731	1735*	1770	1774*	1799	1803*		
1819	1823*	1832	1836*	1861	1865*	1874	1878*	1882	1886*	1892	1896*	1961*		
1964	1968*	1992	1996*	2021*	2023	2027*	2050	2054*	2079*	2081	2085*	2109		
2113*	2137	2141*	2167*	2169	2173*	2196	2200*	2223	2227*					
1357	1379	1814	2361*	2406	2649*									
2309*	2651*	2671*												
449	510*	993	1465	1466										
512*	2428													
459	2669*													
2286*														
443*	461	478*	494*	511*	996*	1135*	1142*	1149*	1156*	1163*	1170*	1177*		
1184*	1191*	1198*	1215*	1218*	1226*	1234*	1242*	1253*	1256*	1268*	1271*	1287*		
1299*	1324*	1336*	1352*	1394*	1424*	1432*	1451*	1456*	1463*	1483*	1492*	1514*		
1516*	1526*	1540*	1661*	1668*	1675*	1682*	1689*	1696*	1703*	1710*	1717*	1724*		
1736*	1738*	1746*	1754*	1762*	1775*	1777*	1790*	1793*	1804*	1807*	1809*	1811*		
1813*	1824*	1837*	1842*	1866*	1879*	1887*	1897*	1900*	2282*	2287*	2295*	2332*		
2377*	2426*	2437*	2478*	2482*	2504*	2511*	2538*	2540*	2550*	2555*	2556*	2608*		
2673*														
506*	1903*	1904*												
2377	2478	2573*												
2332	2437	2538	2550	2555	2565*									
2299	2426	2482	2567*											
431*	512*	952*	997*	1126*	1524	1659*	1991*	2108*	2136*					
1977*	2005*	2036*	2063*	2094*	2122*	2150*	2182*	2209*	2236*	2653*				
514*	548*	582*	616*	650*	684*	721*	761*	801*	874*	947*	1004*	1048*		
1076*	1120*	1135	1136*	1142	1143*	1149	1150*	1156	1157*	1163	1164*	1170		
1171*	1177	1178*	1184	1185*	1191	1192*	1198	1199*	1209*	1215	1216*	1218		
1219*	1226	1227*	1234	1235*	1242	1243*	1247*	1253	1254*	1256	1257*	1268		
1269*	1271	1272*	1281*	1287	1289*	1299	1300*	1324	1325*	1336	1337*	1352		
1353*	1387*	1394	1395*	1424	1425*	1432	1433*	1451	1452*	1456	1457*	1463		
1464*	1483	1484*	1486*	1492	1493*	1508*	1514	1515*	1518*	1526	1527*	1540		
1541*	1552*	1581*	1609*	1653*	1661	1662*	1668	1669*	1675	1676*	1682	1683*		
1689	1690*	1696	1697*	1703	1704*	1710	1711*	1717	1718*	1724	1725*	1731*		
1736	1737*	173*	1739*	1746	1747*	1754	1755*	1762	1763*	1770*	1775	1776*		
1777	1778*	1790	1791*	1793	1794*	1799*	1804	1805*	1807	1808*	1809	1810*		
1811	1812*	1813	1814*	1819*	1824	1825*	1832*	1837	1838*	1842	1843*	1861*		
1866	1867*	1874*	1879	1880*	1882*	1887	1888*	1892*	1897	1898*	1900	1901*		
1964*	1992*	2023*	2050*	2081*	2109*	2137*	2169*	2196*	2223*					
811*	817*	823*	829*	835*	841*	847*	853*	884*	890*	896*	902*	908*		
914*	920*	926*	1134*	1136	1141*	1143	1148*	1150	1155*	1157	1162*	1164		
1169*	1171	1176*	1178	1183*	1185	1190*	1663*	1667	1670*	1674	1677*	1681		
1684*	1688	1691*	1695	1698*	1702	1705*	1709	1712*	1716	1719*				
474*	484	2673												
2540	2548*													
435*	464*	466	474*	480*	520*	521	526*	527	532*	533	538*	539		
544*	545	554*	555	560*	561	566*	567	572*	573	578*	579	588*		
589	594*	595	600*	601	606*	607	612*	613	622*	623	628*	629		
634*	635	640*	641	646*	647	656*	657	662*	663	668*	669	674*		
675	680*	681	690*	691	696*	697	702*	703	708*	709	714*	715		
729*	730*	731	738*	739*	740	747*	748*	749	756*	757*	758	769*		
770*	771	778*	779*	780	787*	788*	789	796*	797*	798	807*	808		

TST212	010204	1864*												
TST213	010242	1877*												
TST214	010262	1885*												
TST215	010302	1895*												
TST3	001206	585*												
TST301	010570	1967*												
TST302	010714	1995*												
TST311	011066	2026*												
TST312	011206	2053*												
TST321	011360	2034*												
TST322	011504	2112*												
TST323	011632	2140*												
TST331	012004	2172*												
TST332	012124	2199*												
TST333	012244	2226*												
TST4	001434	619*												
TST5	001562	653*												
TST6	001710	687*												
TST7	002044	724*												
TTY	=%000005	441*	1814*	1815	1928	1929*	1930	1932	1933*	1934	1936	1944	1952*	1953*
		1954	1955*	2297*	2298*	2331*	2376*	2425*	2436*	2477*	2481*	2537*	2549*	2554*
TYPE	= 000004	2576*	2577*	2578*	2580*	2582*	2588*	2590*	2592*					
		430*	992	1907	2019	2077	2165	2251	2296	2312	2326	2330	2333	2362
		2363	2415	2427	2435	2438	2465	2466	2480	2483	2529	2533	2539	2553
		2606												
WADRA	015050	1289	1301	1314*	1326*	1338*	1359	1382	1403	1413	1419	1441	1464	1471
		1493	1515	2375*	2403*	2620*								
WADRAX	015036	1839	2364	2372	2390	2616*								
WADR8	015076	2631*												
WADR8X	015064	2467	2474	2627*										
WCSRA	015024	520	525*	526	531*	532	537*	538	543*	544	806*	812*	818*	824*
		830*	836*	842*	848*	854*	860*	866*	872*	880	886	892	898	904
		910	916	922	928	934	940	946*	955*	957*	966	975	979*	985*
		999	1009*	1010	1011*	1012	1021*	1022	1031*	1032	1041*	1042	1047*	1127
		1129	1134*	1136	1141*	1143	1148*	1150	1155*	1157	1162*	1164	1169*	1171
		1176*	1178	1183*	1185	1190*	1192	1197*	1199	1204*	1205	1214*	1217*	1225*
		1233*	1241*	1252*	1255*	1267*	1270*	1286*	1293	1298*	1303	1308	1318	1323*
		1330	1335*	1340	1345	1351*	1364	1369	1371	1393*	1408	1412*	1423*	1431*
		1445	1449*	1450*	1455*	1462*	1482*	1491*	1513*	1525*	1531	1539*	1557*	1558
		1563*	1564	1569*	1570	1575*	1576	1580*	1660*	1663	1667*	1670	1674*	1677
		1681*	1684	1688*	1691	1695*	1698	1702*	1705	1709*	1712	1716*	1719	1723*
		1726	1730*	1737*	1745*	1753*	1761*	1769*	1776*	1789*	1792*	1798*	1806*	1808*
		1810*	1812*	1816*	1826*	1827	1829*	1841*	1843	1845	1849*	1858*	1868*	1869
		1871*	1881*	1890*	1898*	1901*	192*	1978	1988	2000*	2006	2014	2031*	2037
		2047	2058*	2064	2074	2089*	2095	2105	2117*	2123	2131	2145*	2151	2161
		2177*	2183	2191	2204*	2210	2220	2231*	2237	2247	2314	2317*	2320	2323*
		2331	2338	2431	2509	2611*								
WCSRB	015052	554	559*	560	565*	566	571*	572	577*	578	807	813	819	825
		831	837	843	849	855	861	867	873*	879*	885*	891*	897*	903*
		909*	915*	921*	927*	933*	939*	945*	956*	961	965*	971*	980	986*
		2434*	2436	2443	2622*									
WDARA	015032	1383	1414	1850	2614*									
WDARB	015060	2625*												
WOBRA	015026	588	593*	594	599*	600	605*	606	611*	612	661*	667*	673*	679*
		1053*	1054	1059*	1060	1065*	1066	1071*	1072	1216*	1224*	1232*	1240*	1254*
		1257	1269*	1272	1586*	1587	1592*	1593	1598*	1599	1604*	1605	1744*	1752*

WDBRAX	015056	1760*	1768*	1783*	1784	1791*	1794	2339	2612*					
WDBRB	015030	656	662	668	674	680	2624*							
WDBRBX	015054	690	696	702	708	714	1219	1227	1235	1243	1262	1277	1739	1747
WIND	015106	1755	1763	1778	2613*									
WRARA	015034	622	627*	628	633*	634	639*	640	645*	646	595*	701*	707*	713*
WRARB	015062	2444	2623*											
WVARA	015040	2360	2464	2636*										
WVARB	015066	728*	729	737*	738	746*	747	755*	756	1086*	1087	1095*	1096	1104*
WVECA	015044	1105	1113*	1114	1119*	1392*	1430*	1619*	1620	1628*	1629	1637*	1638	1646*
WVECB	015072	1647	1652*	1805*	1815*	1825*	1839*	1867*	1880*	1889*	2348*	2349	2615*	
STRACE	012570	768*	769	777*	778	786*	787	795*	796	2453*	2454	2626*		
.	= 017542	2413	2425	2617*										
		2479	2481	2628*										
		953*	1523*	1524*	1535*	1536*	1971*	1999*	2030*	2057*	2088*	2116*	2144*	2176*
		2203*	2230*	2342	2422*	2424*	2496*	2497*	2619*					
		954*	2447	2489*	2491*	2498*	2499*	2630*						
		2282	2287	2293*										
		444*	445	446*	455*	469	486*	493	497	502*	522	528	534	540
		546	556	562	568	574	580	590	596	602*	608	614	624	630
		636	642	648	658	664	670	676	682	692	698	704	710	716
		732	741	750	759	772	781	790	799	809	815	821	827	833
		839	845	851	857	863	869	882	888	894	900	906	912	918
		924	930	936	942	963	968	977	982	1000	1015	1025	1035	1045
		1056	1062	1068	1074	1090	1099	1108	1117	1128	1131	1138	1145	1152
		1159	1166	1173	1180	1187	1194	1201	1207	1221	1229	1237	1245	1259
		1264	1274	1279	1291	1295	1310	1320	1328	1332	1347	1362	1366	1373
		1385	1410	1416	1447	1474	1503	1528	1533	1560	1566	1572	1578	1589
		1595	1601	1607	1623	1632	1641	1650	1665	1672	1679	1686	1693	1700
		1707	1714	1721	1728	1741	1749	1757	1765	1780	1786	1796	1828	1847
		1853	1870	1941	1946	1975	1984	2003	2016	2019	2034	2043	2061	2070
		2077	2092	2101	2120	2133	2148	2157	2165	2180	2193	2207	2216	2234
		2243	2251	2294	2296	2301	2310	2312	2313*	2326	2327*	2330	2331*	2333
		2363	2364*	2415	2416*	2427	2428*	2435	2436*	2438	2466	2467*	2480	2481*
		2483	2484*	2533	2534*	2539	2540*	2542	2546	2547	2566	2595	2598	2603
		2609*	2610	2667*	2676									
.BIT	= 166000	370*	377	378	379	380	381	382	383	384	385	2261	2527	2531
.PR	014776	2544												
.PRF	014716	2563*	2565*	2567*	2568*	2570	2574	2584*	2586	2596*	2597	2600*	2602	2606
.PRL	014674	2609*												
.PTIT	014620	2572	2587	2592*										
.TYPE	010552	2585*	2601											
		2564	2569*											
		1932*	1933	1959*										

200	1904														
200	480	1359	1404	1418	1419	1452	1465	1471	1494	1855	1903	1949	1952	2323	2358
201	2390	2407	2432	2462											
201	1399	1398	1435	1436	1459	1470	2368	2373	2374	2470	2475	2476	2576	2577	2578
201	2391	2390	2457	2461											
201	490														
201	489	497	522	528	524	540	546	556	562	568	574	580	590	596	602
201	600	614	624	630	636	642	648	658	664	670	676	682	692	692	704
201	710	716	732	741	750	759	772	781	790	799	809	815	821	827	833
201	830	845	851	857	863	869	882	888	894	900	906	912	918	924	930
201	930	942	963	968	977	982	1000	1015	1025	1035	1045	1056	1062	1068	1074
201	1090	1099	1108	1117	1128	1131	1138	1145	1152	1159	1166	1173	1180	1187	1194
201	1200	1207	1221	1229	1237	1245	1259	1264	1274	1279	1295	1310	1320	1332	1347
201	1300	1304	1373	1378	1385	1410	1416	1447	1474	1478	1503	1533	1560	1566	1572
201	1570	1589	1595	1601	1607	1623	1632	1641	1650	1665	1672	1673	1686	1693	1700
201	1707	1714	1721	1728	1741	1749	1757	1765	1780	1786	1795	1829	1847	1853	1870
201	1900	1925	1984	2016	2043	2070	2101	2133	2157	2193	2216	2243	2275	2289	2310
201	2300	2429	2460	2502	2510	2528	2552	2595	2598						
210	727	730	736	739	745	748	754	757	767	770	776	779	785	788	794
210	797	865	979	1013	1020	1023	1030	1033	1040	1043	1085	1088	1094	1097	1103
210	1106	1112	1115	1412	1449	1618	1621	1627	1630	1636	1639	1645	1648	1849	1953
210	2398	2395	2433												
210	1370														
210	957	971													
210	2582	2599													
210	468	990	999	1127	1399	1437	1827	1843	1869	1905	1930	2261	2272	2293	2509
210	2527	2531	2586												
210	2517	2419	2485	2487											
210	467	482	991	1400	1421	1438	1454	1480	1498	1507	1545	1547	1844	1906	1931
210	1937	1943	1951	2262	2273	2277	2294	2325	2343	2371	2386	2402	2448	2473	2513
210	2515	2532	2587	2601	2603	2686	2688								
210	1941	1946	2542												
210	484	499	1291	1302	1306	1315	1328	1339	1343	1356	1362	1381	1406	1443	1529
210	1856	1947	1975	1986	2003	2012	2034	2045	2061	2072	2092	2103	2120	2129	2149
210	2159	2180	2189	2207	2218	2234	2245	2263	2268	2271	2315	2321	2328	2334	2345
210	2359	2410	2439	2450	2463	2564	2566	2572	2676						
210	490														
210	464	543	577	611	645	679	713	872	873	945	946	952	985	986	1047
210	1071	1119	1125	1126	1314	1326	1338	1361	1392	1405	1430	1536	1543	1604	1652
210	1558	1659	1805	1825	1839	1867	1880	1889	1991	2108	2354	2365	2366	2380	2458
210	2468	2494	2495	2497	2499	2508	2567	2682	2684						
210	2571	2585	2605												
210	466	479	496	521	527	533	539	545	555	561	567	573	579	589	595
210	601	607	613	623	629	635	641	647	657	663	669	675	681	691	697
210	703	709	715	731	740	749	758	771	780	789	798	808	814	820	826
210	832	838	844	850	856	862	868	881	887	893	899	905	911	917	923
210	929	935	941	962	967	976	981	1024	1034	1044	1055	1061	1067	1073	1089
210	1098	1107	1116	1130	1137	1144	1151	1158	1165	1172	1179	1186	1193	1200	1206
210	1220	1228	1236	1244	1258	1263	1273	1278	1292	1294	1305	1309	1316	1319	1329
210	1331	1342	1346	1363	1365	1372	1384	1407	1409	1415	1420	1444	1446	1473	1476
210	1477	1479	1502	1529	1532	1559	1565	1571	1577	1588	1594	1600	1606	1622	1631
210	1640	1649	1664	1671	1678	1685	1692	1699	1706	1713	1720	1727	1740	1748	1756
210	1764	1779	1785	1795	1846	1852	1950	1976	1983	1987	2004	2013	2015	2035	2042
210	2046	2062	2069	2073	2093	2100	2104	2121	2130	2132	2149	2156	2160	2181	2190
210	2192	2208	2215	2219	2235	2242	2246	2269	2316	2322	2336	2342	2347	2411	2416

	2418	2428	2441	2447	2452	2484	2486	2602													
CMPB	1936	2276																			
COM	719	1082	1615																		
DEC	481	1497	1506	1544	1546	1942	2324	2370	2385	2401	2472	2514	2685	2687							
ENT	429																				
HALT	445	470	483	1974	1980	2002	2008	2020	2033	2039	2060	2056	2078	2091	2097						
	2119	2125	2147	2153	2166	2179	2185	2206	2212	2233	2239	2252	2327	2335	2346						
	2420	2440	2451	2488	2518	2543															
INC	2309	2389	2512	2530																	
INCB	2279	2285	2596	2600																	
IOY	430																				
JMP	448	449	450	451	452	453	457	459	472	993	1001	1549	1914	2299	2311						
	2421	2430	2689																		
JSR	511	996	1135	1142	1149	1156	1163	1170	1177	1184	1191	1198	1215	1218	1226						
	1234	1242	1253	1256	1268	1271	1287	1299	1324	1336	1352	1394	1424	1432	1451						
	1456	1463	1483	1492	1514	1516	1526	1540	1661	1668	1675	1682	1689	1696	1703						
	1710	1717	1724	1736	1738	1746	1754	1762	1775	1777	1790	1793	1804	1807	1803						
	1811	1813	1824	1837	1842	1866	1879	1887	1897	1900	1910	2282	2287	2332	2377						
	2426	2437	2478	2482	2538	2540	2550	2555	2673												
MOV	461	465	471	474	475	476	488	489	495	510	512	520	525	526	531						
	532	537	538	544	554	559	560	565	566	571	572	578	588	593	594						
	599	600	605	606	612	622	627	628	633	634	639	640	646	656	661						
	662	667	668	673	674	680	690	695	696	701	702	707	708	714	718						
	726	728	729	735	737	738	744	746	747	753	755	756	766	768	769						
	775	777	778	784	786	787	793	795	796	806	807	812	813	818	819						
	824	825	830	831	836	837	842	843	848	849	854	855	860	861	866						
	867	879	880	885	886	891	892	897	898	903	904	909	910	915	916						
	921	922	927	928	933	934	939	940	953	954	955	956	960	961	966						
	974	975	980	995	997	1009	1010	1011	1012	1019	1021	1022	1029	1031	1032						
	1039	1041	1042	1053	1054	1059	1060	1065	1066	1072	1081	1084	1086	1087	1093						
	1095	1096	1102	1104	1105	1111	1113	1114	1129	1134	1136	1141	1143	1148	1150						
	1155	1157	1162	1164	1169	1171	1176	1178	1183	1185	1190	1192	1197	1199	1205						
	1214	1216	1217	1219	1224	1225	1227	1232	1233	1235	1240	1241	1243	1252	1254						
	1255	1257	1262	1267	1269	1270	1272	1277	1286	1288	1293	1298	1300	1303	1309						
	1313	1318	1323	1325	1330	1335	1337	1340	1345	1351	1355	1357	1358	1360	1364						
	1369	1371	1379	1380	1383	1393	1395	1401	1402	1403	1408	1414	1422	1423	1431						
	1433	1439	1440	1441	1442	1445	1450	1455	1462	1464	1466	1467	1472	1482	1491						
	1493	1495	1496	1500	1501	1513	1515	1523	1524	1525	1531	1535	1539	1542	1557						
	1558	1563	1564	1569	1570	1575	1576	1580	1586	1587	1592	1593	1598	1599	1605						
	1614	1617	1619	1620	1626	1628	1629	1635	1637	1638	1644	1646	1647	1660	1663						
	1667	1670	1674	1677	1681	1684	1688	1691	1695	1698	1702	1705	1709	1712	1716						
	1719	1723	1726	1730	1737	1739	1744	1745	1747	1752	1753	1755	1760	1761	1763						
	1768	1769	1776	1778	1783	1784	1789	1791	1792	1794	1798	1806	1808	1810	1812						
	1814	1815	1816	1826	1829	1839	1841	1845	1850	1851	1858	1868	1871	1881	1890						
	1898	1901	1908	1927	1928	1929	1932	1933	1948	1954	1955	1956	1961	1962	1969						
	1970	1971	1972	1977	1978	1979	1981	1982	1988	1990	1997	1998	1999	2000	2005						
	2006	2007	2009	2010	2014	2018	2021	2028	2029	2030	2031	2036	2037	2038	2040						
	2041	2047	2049	2055	2056	2057	2058	2063	2064	2065	2067	2068	2074	2076	2079						
	2086	2087	2088	2039	2094	2095	2096	2098	2099	2105	2107	2114	2115	2116	2117						
	2122	2123	2124	2126	2127	2131	2135	2136	2142	2143	2144	2145	2150	2151	2152						
	2154	2155	2161	2163	2167	2174	2175	2176	2177	2182	2183	2184	2186	2187	2191						
	2195	2201	2202	2203	2204	2209	2210	2211	2213	2214	2220	2222	2228	2229	2230						
	2231	2236	2237	2238	2240	2241	2247	2249	2264	2265	2267	2270	2280	2281	2286						
	2290	2302	2303	2304	2305	2306	2307	2308	2313	2317	2318	2319	2331	2337	2338						
	2339	2341	2344	2348	2349	2350	2360	2361	2364	2367	2372	2375	2376	2378	2381						
	2382	2383	2384	2387	2388	2390	2396	2397	2398	2400	2403	2404	2405	2406	2408						

	2409	2412	2413	2414	2422	2424	2425	2431	2434	2436	2442	2443	2444	2446	2449
	2453	2454	2464	2467	2469	2474	2477	2479	2481	2489	2491	2492	2493	2496	2498
	2500	2503	2507	2516	2534	2537	2549	2554	2563	2569	2570	2573	2574	2584	2607
	2670	2671	2672	2674	2675	2683									
MOV8	1002	1204	1550	1938	1939	1944	2278	2297	2536	2565	2568	2575	2604		
NEG	1396	1434	1468	2351	2455										
NOP	471	958	972	998	1911	1912	1913	2677	2678	2679	2680	2681			
RESET	462	519	553	587	621	655	689	994	2669						
ROL	491	492	2369	2471	2588	2590	2592								
ROLB	2589	2591	2593												
ROR	2391	2392	2393	2394	2580	2582									
RTI	1957	2283	2291	2544											
RTS	478	494	2295	2504	2511	2556	2608								
SUB	463	1382	1413	1453	1499	2535									
SWAB	2352	2456													
TRAP	428														
TST	477	1014	1289	1301	1353	1377	1505	1530	2266	2288	2314	2320	2340	2355	2379
	2423	2445	2459	2490	2501	2541	2551								
TSTB	1934	1940	1945	2274	2594	2597									
WAIT	1973	2001	2032	2059	2090	2118	2146	2178	2205	2232					
.ASCII	2636														
.ASCIZ	2020	2078	2166	2252	2297	2313	2327	2331	2334	2364	2416	2428	2436	2439	2467
	2481	2484	2534	2540											
.BLKW	2609														
.ENABL	1	370													
.END	2692														
.ENDC	444	525	559	593	627	661	695	1053	1586	1667	1674	1681	1688	1695	1702
	1709	1716	1934	1958	1964	1968	1986	1990	1996	2009	2018	2023	2027	2045	2049
	2050	2054	2072	2076	2081	2095	2103	2107	2113	2126	2135	2137	2141	2159	2163
	2173	2186	2195	2196	2200	2218	2222	2223	2227	2245	2249	2261	2262	2300	2531
	2533	2541	2545	2557	2585	2611									
.EVEN	2020	2078	2166	2252	2297	2313	2327	2331	2334	2364	2416	2428	2436	2439	2467
	2481	2484	2534	2540											
.IF	444	519	553	587	621	655	689	1053	1586	1663	1670	1677	1684	1691	1698
	1705	1712	1930	1948	1964	1981	1992	1996	2009	2023	2040	2050	2067	2081	2098
	2109	2113	2126	2137	2154	2169	2173	2186	2196	2213	2223	2240	2261	2292	2527
	2531	2540	2544	2557	2573	2610									
.IFF	1667	1670	1677	1684	1691	1698	1705	1712	1958	1964	1986	1987	1988	1992	1996
	2009	2014	2023	2045	2046	2047	2050	2054	2067	2073	2074	2081	2085	2098	2104
	2105	2109	2113	2126	2131	2137	2159	2160	2161	2169	2173	2186	2191	2196	2200
	2213	2219	2220	2223	2245	2246	2247	2261	2527	2545	2557				
.IFT	1988	2018	2047	2074	2105	2135	2161	2195	2220	2247					
.IFTF	1987	2013	2046	2073	2104	2130	2160	2190	2219	2246					
.IIF	377	378	379	380	381	382	383	384	385	811	817	823	829	835	841
	847	884	890	896	902	908	914	920	1134	1141	1148	1155	1162	1169	1176
	1183	1663	1670	1677	1684	1691	1698	1705	1712	2282	2287	2301	2546	2547	2610
.IRP	2264	2267	2270												
.LIST	1	370	444	445	501	514	518	548	552	582	586	616	620	650	654
	684	688	721	725	761	765	801	805	811	817	823	829	835	841	847
	853	874	878	884	890	896	902	908	914	920	926	947	951	994	1004
	1008	1048	1052	1076	1080	1120	1124	1134	1136	1141	1143	1148	1150	1155	1157
	1162	1164	1169	1171	1176	1178	1183	1185	1190	1192	1199	1209	1213	1216	1219
	1227	1235	1243	1247	1251	1254	1257	1269	1272	1281	1285	1288	1300	1325	1337
	1353	1387	1391	1395	1425	1429	1433	1452	1457	1461	1464	1484	1486	1490	1493
	1508	1512	1515	1518	1522	1527	1541	1550	1552	1556	1581	1585	1609	1613	1653
	1657	1662	1663	1669	1670	1676	1677	1683	1684	1690	1691	1697	1698	1704	1705

	1711	1712	1718	1719	1725	1731	1735	1737	1739	1747	1755	1763	1770	1774	1776
	1778	1791	1794	1799	1803	1805	1808	1810	1812	1814	1819	1823	1825	1832	1836
	1838	1843	1861	1865	1867	1874	1878	1880	1882	1886	1888	1892	1896	1898	1901
	1919	1961	1964	1968	1992	1996	2020	2021	2023	2027	2050	2054	2078	2079	2081
	2085	2109	2113	2137	2141	2166	2167	2169	2173	2196	2200	2223	2227	2252	2253
	2297	2302	2313	2327	2331	2334	2364	2416	2428	2436	2439	2457	2481	2484	2519
	2534	2540	2557	2611											
.MACRO	1	444													
.MCALL	370	444	514	548	582	616	650	684	721	761	801	874	947	1004	1048
	1076	1120	1209	1247	1281	1387	1425	1457	1486	1508	1518	1552	1581	1609	1653
	1731	1770	1799	1819	1832	1861	1874	1882	1892	1964	1992	2023	2050	2081	2109
.NLIST	2137	2169	2196	2223											
	1	370	444	445	501	514	518	548	552	582	586	616	620	650	654
	684	689	721	725	761	765	801	805	811	817	823	829	835	841	847
	853	874	878	884	890	896	902	908	914	920	926	947	951	994	1004
	1008	1048	1052	1076	1080	1120	1124	1134	1136	1141	1143	1148	1150	1155	1157
	1162	1164	1169	1171	1176	1178	1183	1185	1190	1192	1199	1209	1213	1216	1219
	1227	1235	1243	1247	1251	1254	1257	1269	1272	1281	1285	1288	1300	1325	1337
	1353	1387	1391	1395	1425	1429	1433	1452	1457	1461	1464	1484	1486	1490	1493
	1508	1512	1515	1518	1522	1527	1541	1550	1552	1556	1581	1585	1609	1613	1653
	1657	1662	1663	1669	1670	1676	1677	1683	1684	1690	1691	1697	1698	1704	1705
	1711	1712	1718	1719	1725	1731	1735	1737	1739	1747	1755	1763	1770	1774	1776
	1778	1791	1794	1799	1803	1805	1808	1810	1812	1814	1819	1823	1825	1832	1836
	1838	1843	1861	1865	1867	1874	1878	1880	1882	1886	1888	1892	1896	1898	1901
	1919	1961	1964	1968	1992	1996	2020	2021	2023	2027	2050	2054	2078	2079	2081
	2085	2109	2113	2137	2141	2166	2167	2169	2173	2196	2200	2223	2227	2252	2253
	2297	2302	2313	2327	2331	2334	2364	2416	2428	2436	2439	2467	2481	2484	2519
	2534	2540	2557	2611											
.PAGE	408	444	548	582	616	650	684	718	761	801	841	874	914	947	1048
	1076	1120	1169	1209	1247	1281	1387	1425	1508	1581	1609	1653	1698	1731	1770
	1799	1882	1902	1992	2021	2050	2079	2109	2137	2167	2196	2223	2302	2557	2611
.REM	1														
.REPT	445	811	884	1134	1663										
.SBTTL	444	501	514	548	582	616	650	684	721	761	801	874	947	994	1004
	1048	1076	1120	1209	1247	1281	1387	1425	1457	1486	1508	1518	1550	1552	1581
	1609	1653	1731	1770	1799	1819	1832	1861	1874	1882	1892	1919	1964	1992	2023
	2050	2081	2109	2137	2169	2196	2223	2253	2302	2519	2557	2611			
.TITLE	370														

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

* DZDAAA.SEO/SOL/CRF/PAGNUM/NL:TOC=STEVE.SML,DZDAAA.P11
RUN-TIME: 27 37 5 SECONDS
RUN-TIME RATIO: 379/70=5.3
CORE USED: 27K (53 PAGES)

