COMPUTE! ISSUE 1 / FALL 1979 / PAGE 32

Universal 6502 Memory Test

Carl W. Moser

This article contains a memory test program which tests RAM memory in various 6502 based systems. This test was developed after using several tests which did not perform a complete test. The problem areas were untested chip selects and address line inputs.

The program performs two tests:

Test 1: Tests memory cells for storage retention, and open, shorted, or non-functioning data and Ao-An address lines. This is done by writing 00 011 ... FF 00 011 ... FF continually throughout the memory range for the first pass. When this has been written, it is checked to validate the data. On the next pass 01 02 ... FF 00 011 ... FF is written and checked. This continues for 256 (hex FF) passes until all possible combinations of bit patterns have been used.

Test 2: Tests the RAM chip select inputs. This is the same as test 1 except data 00 01 ... F2 00 01 ... F2 is used. The purpose of this test is to test the remaining A_8-A_{15} address lines. Listings 1 (originating at memory address $0002) and 2 (originating at $0800) contain the source of the memory test program. The reason for these two listings is that not all 6502 microcomputers have RAM at a common address from which the memory test program can execute. To determine which listing is appropriate for your system, consult, table A. Next enter the object code from the appropriate listing, and then configure the I/O for your system, also from table A.

Enter the start address and end address of the memory range to be tested as described in table B. Execution begins with test 1 at $0002 for Listing 1 and $0800 for Listing 2.

If an error occurs, it will be outputted in the following format:

Address	Test Pattern	Error
xxxx	yy	zz

Note: This program performs a lengthy but exhaustive rest of RAM memory. It takes approximately 38 seconds per 1K of memory for each test 1 and test 2.

When test 1 runs to completion, a break instruction will be executed to enter your systems monitor program. Register A will contain E1 indicating end of test 1. To execute test 2, simply continue execution by typing G to your monitor.

If errors occur, they will be of the same form as described above. When test 2 has run to completion, a break instruction will again transfer control to your monitor and register A will contain E2 signifying the end. To continue execution again at test 1, simply type G. The start and end address range is not altered by the memory test program.

If errors occurred in test 2 but not in test 1, you can safely assume a chip select malfunction (possible stuck in enable state or malfunction with circuitry which generates the chip select or an address line other than A_0-A_7. Usually a number of errors will occur in test 1 when the fault is a single defective address input, data input, or data output.

If a continuous sequence of addresses with errors occur, the problem is likely to be an open data input or a data output stuck at '1' or '0.'

If errors occur every 2nd, 4th, 8th, 16th or some power of 2 address sequence, check for defective address inputs as follows:

Data bit with error	Check address input	Data bit with error	Check address input
D_0	A_0 or A_8	D_4	A_4 or A_{12}
D_1	A_1 or A_9	D_5	A_5 or A_{13}
D_2	A_2 or A_{10}	D_6	A_6 or A_{14}
D_3	A_3 or A_{11}	D_7	A_7 or A_{15}

If, for example, you are checking 2102's (1 × 1K) and are specifying a 4K range of memory and an error common to the whole range occurs, the problem is likely to be in the power leads, defective data or address buffers, stuck at '0' address inputs, stuck at '0' data inputs, or stuck at '0' data outputs.

In all of the above, you may have to examine the various memory error patterns for some similarity in order to isolate the defective component. This is especially true of the 1 × 1K 2102, and 1 × 16K 4116 memory chips where each chip is devoted to a particular data lead (D_0-D_7).

```
TEST 2   TEST 1                0010 ; MCS 6502 MEMORY TEST
                               0080 ; ZERO PAGE LOCATIONS
                               0090 ADDRS      .DE 0 ; 2 BYTES - ADDRESS OF MEMORY
                               0100
                               0140            .BA $0002 OR .BA $0800
0800- A2 00     0002- A2 00    0160 MEM<TEST LDX #$00
0802- BE E2 08  0004- 8E E4 00 0170            STX TEST<TYPE ; TEST 1
0805- 20 1B 08  0007- 20 ID 00 0180            JSR TEST<PGM
0808- A9 E1     006A- A9 E1    0190            LDA #$E1
080A- 00        000C- 00       0200            BRK
080B- EA        000D- EA       0210            NOP
080C- EA        000E- EA       0220            NOP
080D- EE E2 08  000F- EE E4 00 0230            INC TEST<TYPE  ; TEST 2
0810- 20 18 08  0012- 20 ID 00 0240            JSR TEST<PGM
0813- A9 E2     0015- A9 E2    0250            LDA #$E2
0815- 00        0017- 00       0260            BRK
0816- EA        0018- EA       0270            NOP
0817- EA        0019- EA       0260            NOP
0818- 4C 00 08  001A- 4C 02 00 0290            JMP MEM<TEST
                               0300 ;
081B- 20 C9 08  001D- 20 CB 00 0310 TEST<PGM JSR CRLF
081B- A0 00     0020- A0 00    0320            LDY #$00 ; PATTERN REGISTER
0820- A2 00     0022- A2 00    0330            LDX #$00
0822- BE E1 08  0024- BE E3 00 0340            STX TEST<PATRN
0825- 4C 2B 08  0027- 4C 30 00 0350            JMP NX<PASS
                               0360 ;
0825- EE E1 08  002A- EE E3 00 0370 NX<PATRN INC TEST<PATEN
0826- D0 01     002D- D0 01    0380            BNE NX<PASS
082D- 60        002F- 60       0390            RTS
082E- AC E1 08  0030- AC E3 00 0400 NX<PASS    LDY TEST<PATRN
0831- 20 9F 08  0033- 20 A1 00 0410            JSR INI<ADDRS
0834- 98        0036- 98       0420 LOOP1      TYA
0835- 81 00     0037- 81 00    0430            STA (ADDRS, X) ; STORE PATTERN
```

```
0837- C1 00     0039- C1 00     0440            CMP (ADDRS, X) ; CHECK
0839- F0 03     003B- F0 03     0450            BEQ NO<ERR1
083B- 20 81 08  003D- 20 83 00  0460            JSR ERROR ; ADDRS, R(A), (ADDRS, X)
083E- 20 6E 08  0040- 20 70 00  0470 NO<ERR1    JSR INC<ADDRSC
0841- F0 06     0043- P0 06     0480            BEQ CK<PATRN
0843- 20 61 08  0045- 20 63 00  0490            JSR INC<RY
0846- 4C 34 08  0048- 4C 36 00  0500            JMP LOOP 1
                                0510 ;
0849- AC E1 08  004B- AC E3 00  0520 CK<PATRN LDY TEST<PATRN
084C- 20 9F 08  004E- 20 A1 00  0530            JSR INI<ADDRS ; INITIALISE ADDRS
084F- 98        0051- 98        0540 LOOP2      TYA
0850- C1 00     0052- C1 00     0550            CMP (ADDRS, X)0852- F0 03     0054- F0 03     0560              BEQ NO <ERR2
0854- 20 81 08  0056- 20 83 00  0570            JSR ERROR ; ADDRS, R(A), (ADDRS, X)
0857- 20 61 08  0059- 20 63 00  0580 NO < ERR2 JSR INC < RY
085A- 20 6E 08  005C- 20 70 00  0590            JSR INC < ADDRSC
085D- D0 F0     005F- D0 F0     0600            BNE LOOP2
085F- F0 C7     0061- F0 C7     0610            BEQ NX < PATRN
0861- C8        0063- C8        0640 INC < RY INY
0862- AD E2 08  0064- AD E4 00  0650            LDA TEST < TYPE
0865- F0 06     0067- F0 06     0660            BEQ EXIT1
0867- C0 F3     0069- C0 F3     0670            CPY #$F3 ; RESET R(Y) TO CHECK
0869- 90 02     006B- 90 02     0680            BCC EXIT1           CHIP SELECTS
086B- A0 00     006D- A0 00     0690            LDY #$00
086D- 60        006F- 60        0700 EXIT1      RTS
                                0710 ;
                                0720 ;
086E- E6 00     0070- E6 00     0730 INC<ADDRSC INC *ADDRS
0870- D0 02     0072- D0 02     0740            BNE SKIP < HI
0872- E6 01     0074- E6 01     0750            INC *ADDRS + $01
0874- AD DF 08  0076- AD E1 00  0760 SKIP<HI    LDA END
0877- C5 00     0079- C5 00     0770            CMP *ADDRS
0879- D0 05     007B- D0 05     0780            BNE EXIT2
087B- AD E0 08  007D- AD E2 00  0790            LDA END + $01
087E- C5 01     0080- C5 01     0800            CMP *ADDRS + $01
0880- 60        0082- 60        0810 EXIT 2     RTS
                                0840 ; OUTPUT THE ERROR; ADDRESS, PATTERN, ERROR
0881- 48        0083- 48        0850 ERROR      PHA
0882- A5 01     0084- A5 01     0860            LDA *ADDRS + $01
0884- 20 AA 08  0086- 20 AC 00  0870            JSR TBYT ; OUTPUT ADDRS HI
0887- A5 00     0089- A5 00     0880            LDA *ADDRS
0889- 20 AA 08  008B- 20 AC 00  0890            JSR TBYT ; OUTPUT ADDRS LO
088C- 20 D4 08  008E- 20 D6 00  0900            JSR SPACE2
088F- 68        0091- 68        0910            PLA
0890- 20 AA 08  0092- 20 AC 00  0920            JSR TBYT ; OUTPUT PATTERN
0893- 20 D4 08  0095- 20 D6 00  0930            JSR SPACE2
0896- A1 00     0098- A1 00     0940            LDA (ADDRS, X)
0898- 20 AA 08  009A- 20 AC 00  0950            JSR TBYT ; OUTPUT ERROR IN MEMORY
089B- 20 C9 08  009D- 20 CB 00  0960            JSR CRLF
089E- 60        00A0- 60        0970            RTS
                                0980 ;
                                0990 ;
                                1000 ; INITIALIZE ADDRS WITH START
089F- AD DD 08  00A1- AD DF 00  1010 INI<ADDRS  LDA START
08A2- 85 00     00A4- 85 00     1020            STA *ADDRS
08A4- AD DE 08  00A6- AD E0 00  1030            LDA START + $01
08A7- 85 01     00A9- 85 01     1040            STA *ADDRS + $01
```

```
08A9- 60        00AB- 60        1050            RTS
                                1060 ;
                                1070 ;
                                1080 ; ROUTINE TO OUTPUT A BYTE
08AA- 48        00AC- 48        1090 TBYT       PHA
08AB- 4A        00AD- 4A        1100            LSR A
08AC- 4A        00AE- 4A        1110            LSR A
08AD- 4A        00AF- 4A        1120            LSR A
08AF- 20 B3 08  00B1- 20 B5 00  1140            JSR NIBBLE
08B2- 68        00B4- 68        1150            PLA
08B3- 29 0F     00B5- 29 0F     1160 NIBBLE     AND #$0F
08B5- 09 30     00B7- 09 30     1170            ORA #$30
08B7- C9 3A     00B9- C9 3A     1180            CMP #$3A
08B9- 90 02     00BB- 90 02     1190            BCC WRITE
08BB- 69 06     00BD- 69 06     1200            ADC #$06
                                1210
                                1220 ; ROUTINE TO WRITE AN ASCII CHAR.
08BD- 8C E3 08  00BF- 8C E5 00  1230 WRITE      STY SAVEY
08C0- EA        00C2- EA        1240 ROM.LINK   NOP
08C1- EA        00C3- EA        1250            NOP
08C2- EA        00C4- EA        1260            NOP
08C3- EA        00C5- EA        1270            NOP
08C4- EA        00C6- EA        1280            NOP
08C5- AC E3 08  00C7- AC E5 00  1290            LDY SAVEY
08C8- 60        00CA- 60        1300            RTS
                                1310
                                1320 ; ROUTINE TO OUTPUT CRLF
08C9- A9 0D     00CB- A9 0D     1330 CRLF       LDA #$0D
08CB- 20 BD 08  00CD- 20 BF 00  1340            JSR WRITE
08CE- A9 0A     00D0- A9 0A     1350            LDA #$0A
08D0- 20 BD 08  00D2- 20 BF 00  1360            JSR WRITE
08D3- 60        00D5- 60        1370            RTS
                                1380
                                1390 ; SPACE2 = OUTPUT 2 SPACES
                                1400 ; SPACE  = OUTPUT 1 SPACE
08D4- 20 D7 08  00D6- 20 D9 00  1410 SPACE2     JSR SPACE
08D7- A9 20     00D9- A9 20     1420 SPACE      LDA #’
08D9- 20 BD OB  00DB- 20 BF 00  1430            JSR WRITE
08DC- 60        00DE- 60        1440            RTS
                                1450
                                1460
08DD-           00DF-           1470 START      .DS 2 ;USER ENTERS START OF MEMORY RANGE
08DF-           00E1-           1480 END        .DS 2 ;USER ENTERS END OF MEMORY RANGE
                                1490
08E1-           00E3-           1500 TEST < PATRN .DS 1        ;CURRENT TEST PATTERN
08E2-           00E4-           1510 TEST < TYPE  .DS 1        ; = 1, 2 FOR TEST TYPE
08E3-           00E5-           1520 SAVEY           .DS 1        ; SAVE R (Y)
                                1530
                                1540
                                1550 END.PGM         .EN
```

Statement 140: $0002 For Test 1 $0800 For Test 2

TABLE A

Computer	Use Listing	Enter at ROM LINK: 00C2 for Listing 1 08C0 for Listing 2
PET	2	20 D2 FF
APPLE II	2	09 80 20 ED FD
SYM	1 or 2	20 63 A6
KIM	1	20 A0 1E
TIM	1 or 2	20 C6 72
OSI 65D	2	20 0B FE
Western Data Systems	2	20 A5 FC
ATARI	?	?
AIM	?	?
Super KIM	?	?

TABLE B

	Listing 1	Listing 2
Start Address lo	00DF	08DD
Start Address hi	00E0	08DE
End Address lo	00E1	08DF
End Address hi	00E2	08E0
Execution Address	0002	0800

Universal 6502 Memory Test
EASTERN HOUSE SOFTWARE
Carl W. Moser
3239 Linda Drive
Winston-Salem, NC 27106

- Back to previous page
- See this article as it appeared in the magazine
- View this issue's table of contents