OSI Memory Test in BASIC

William L. Taylor
246 Flora Road
Leavittsburg, OH 44430

All memory tests are not alike. This one features an extensible, BASIC language implementation.

Have you experienced the complete failure of your favorite program lately? Have you reloaded it into the machine only to have it bomb over and over again? Well, I have, and many times! This could be caused by a bug in the program, but if the program has run before and now bombs there must be something wrong in the hardware. This usually means that there is a reclusive bug hidden somewhere in those many K's of RAM.

How do you find this reclusive bug? If you have a machine code monitor and loader, you could load the memory and step through the program checking for errors. You might also load a diagnostic program to test the memory. "OK" you say, "but I don't have a machine code monitor. My machine has only BASIC in ROM. What do I do to check for these bugs in my machine? I have no means to get at these bugs in my machine with this BASIC only!"

Well take heart, all is not lost. I have had this same experience. Felt the same wrath, of the same bug in those many K's of RAM, that you are feeling now! From this experience I made a decision. I decided to prevent this from doing me in over and over again. My solution to the bug-in-memory caper was to write a diagnostic program, in BASIC, to check the memory of the BASIC-in-ROM only machine.

The program that I have written will load memory with an inital value stored in the D variable, between the address limits P1 and P2. The program increments the D variable from its initial value to 255 decimal. This represents all combinations of bits that can be stored in a memory location. After the bits are stored, the program compares the data bits in memory to the initial value that was stored there and, if they are not the same, a report will be printed out to the terminal.

I have written the program to request page numbers for the starting and ending addresses. This could be changed to use decimal equivalents if the reader wishes. The starting address is contained in variable P1 at line 700. The ending address is contained in P2 at line 710. The contents of both variables are multiplied by 256 to obtain the decimal equivalent of the page numbers. Line 720 is the inital value of the data and is usually set to 0.

At line 750 the program is told to load the limits of memory between P1 and P2 via a FOR-NEXT loop. At line 760 the data bits are POKEd into memory. Line 785 looks at the data in the memory location that was previously stored. At line 790 I compare the data stored in memory against the data in variable D to see if the two are equal. The next byte is loaded and compared at line 800.

Line 825 increments the data value in the D variable. Line 830 checks the D variable to see if 255 decimal has been reached and, if not, executes a return loop through the program. Line 840 reports the results of the memory test.

This program was written in MicroSoft BASIC for the OSI Challenger. It should run under other BASICs with minor modifications. The program will be of interest to users of machines with BASIC in ROM and others who want a simple way to test memory. The program is some what slow, but this a very small price to pay for the ease of operation. Good luck and good memory testing.

```
650 REM MEMORY TEST  BY W.L. TAYLOR 1/2/79
660 PRINT " ******MEMORY TEST****** ":PRINT
665 PRINT " ENTER STARTING PAGE AND ENDING PAGE":PRINT
700 INPUT " STARTING PAGE ";P1
710 INPUT " ENDING PAGE   ";P2
720 D=0
730 LET A=P1*256
740 LET B=P2*256
750 FOR C= A TO B
760 POKE C,D
770 E=PEEK (C)
780 IF E<>D THEN PRINT " BAD DATA BYTE AT";C
790 IF E<>D THEN END
800 NEXT C
810 D=D+1
820 IF D<256 THEN 750
830 IF D=256 THEN PRINT " TEST COMPLETE WITH NO BAD DATA BITS
    DETECTED":PRINT
840 END
```