Tricky Code that Always Skips#

General Information

Author: Bob Sander-Cederlof
Assembler: generic
Published: APPLE Assembly Line 03/82
Download: http://www.txbobsc.com/aal/

All microprocessors have an instruction which does nothing, usually called "NOP". The 6502 is no exception.

In spite of appearances, an instruction which does nothing can be quite useful. However, this article is about another kind of instruction, which does ALMOST nothing.

Some microprocessors have this kind, which do nothing except skip over one or more bytes. That is, they act like a very short forward jump. The advantage over using an actual jump or branch instruction is in memory saved. Relative branches on the 6502 take two bytes of memory; jumps take three. A skip-one or skip-two instruction would take only one byte, IF the 6502 had such.

IF? Well, you certainly do not see an instruction like this among the 56 in any of the books, do you?

However, if you disassemble things like DOS, Applesoft ROMs, and printer interface ROMs, you will find tricky ways to skip with only one byte. For example, in many Apple printer interfaces, the first three bytes look like this:

```
C100-  18         CLC
C101-  B0 38      BCS $C13B
```

Now isn't that silly: to clear carry, and then to use BCS to branch if it is not clear!? No, the BCS is just being used to skip over the $38 stored in $C102. If you enter the code at $C102, that $38 is a SEC instruction. Thus, depending on whether you entered at $C100 or $C102, carry is clear or set respectively. The BCS opcode byte is being used as a skip-one opcode.

Another kind of skip is found in various places inside your Apple. You might find the BIT instruction used this way. In fact, it seems to me that I run across BIT being used as a skip-one or skip-two instruction more often than I see it used to test bits! Here is an example from Applesoft ROMs:

```
E196-  A2 78     LDX #$6B         "BAD SUBSCRIPT" MSG
E198-  2C A2 35  BIT $35A2    TRICK: BIT SKIPS OVER 2
E19B-  4C 12 D4  JMP $D412    PRINT ERROR MESSAGE
```

The code should really look like this:

```
E196-  A2 78      LDX #$6B           "BAD SUBSCRIPT" MSG
E198-  2C     .HS 2C             SKIP NEXT TWO BYTES
E199-  A2 35      LDX #$35           "ILLEGAL QUANTITY" MSG
E19B-  4C 12 D4   JMP $D412          PRINT ERROR MESSAGE
```

You have to be a little careful about what you skip over. The BIT instruction is actually executed, and so status flags Z, N, and V are possibly changed. Also, the two bytes skipped over represent a memory address to the BIT opcode; that memory location will be accessed. No problem, unless it just happens to be an address in the range of the I/O addresses (from $C000 to $CFFF). If it does, something strange might occur, like turning on a disk drive....

If you remember my article about the "So-Called Unused Opcodes", from about a year ago, there are some REAL skip-one and skip-two instructions. They do not modify any status bits, and they do not

reference any memory addresses. I would recommend using ".HS 3C" rather than ".HS 2C" for this reason. "3C" is not a defined or supported opcode, but it apparently is built-in to all existing 6502's. (No guarantee here...test your own before you make a big commitment.)

If you want to skip only one byte, you can use the other BIT form ($24); it works on a zero page address, which will not bother any I/O addresses. If you don't want to modify any status bits, try ".HS 34".