NOT Operation

The **NOT instruction** is used to invert or toggle all the bits in an operand. This operation is also known as taking the one's complement of the operand. Here's how it works:

NOT reg: This form of the NOT instruction operates on a register. It inverts all the bits in the specified register.

```
215 mov al, 11110000b ; AL = 11110000b (F0h in hexadecimal)
216 not al            ; AL = 00001111b (0Fh in hexadecimal)
```

NOT mem: This form of the NOT instruction operates on a memory location. It inverts all the bits in the value stored at that memory location.

```
218 mov byte ptr [ebx], 10101010b ; Store 10101010b at the memory location pointed to by EBX
219 not byte ptr [ebx]            ; Invert the bits at that memory location
```

In both examples, the NOT instruction flips all the bits. In the first example, it's applied to the AL register, and in the second example, it's applied to a byte stored in memory via the EBX register.

Flags: The NOT instruction does not affect any of the CPU flags. It simply performs the bitwise inversion without changing the status flags like Zero Flag, Sign Flag, etc.