

**Overflow Flag** The Overflow flag is set when the result of a signed arithmetic operation overflows or underflows the destination operand. For example, from Chapter 1 we know that the  
largest possible integer signed byte value is 127; adding 1 to it causes overflow:  
mov al,+127  
add al,1 ; OF = 1

**Sign Flag** The Sign flag is set when the result of a signed arithmetic operation is negative. The  
next example subtracts a larger integer (5) from a smaller one (4):  
mov eax,4  
sub eax,5 ; EAX = -1, SF = 1

**Parity** The Parity flag (PF) is set when the least significant byte of the destination has an even  
number of 1 bits. The following ADD and SUB instructions alter the parity of AL:  
mov al,10001100b  
add al,00000010b ; AL = 10001110, PF = 1  
sub al,10000000b ; AL = 00001110, PF = 0

OV (overflow flag), UP (direction flag), EI(interrupt flag), PL (sign flag), ZR (zero flag), AC (auxiliary carry), PE (parity flag), and CY(carry flag).

**Auxiliary Carry** The Auxiliary Carry (AC) flag indicates a carry or borrow out of bit 3 in the  
destination operand. It is primarily used in binary coded decimal (BCD) arithmetic, but can be  
used in other contexts. Suppose we add 1 to 0Fh. The sum (10h) contains a 1 in bit position 4  
that was carried out of bit position 3:  
mov al,0Fh  
add al,1 ; AC = 1  
Here is the arithmetic:  
0 0 0 0 1 1 1 1  
+ 0 0 0 0 0 0 0 1  
------------------  
0 0 0 1 0 0 0 0

Ko thay đổi bit 3 4 = >về 0

The Interrupt flag controls the way the CPU responds to external (hardware) interrupts. If the  
Interrupt flag is set, interrupts are enabled; if the flag is clear, interrupts are disabled. The STI (set  
interrupt) instruction enables interrupts; the CLI (clear interrupt) instruction disables interrupts.

The **direction flag** is a [flag](https://en.wikipedia.org/wiki/Flag_field) that controls the *left-to-right* or *right-to-left* direction of [string](https://en.wikipedia.org/wiki/String_(computer_science)) processing

