Flags

A flag is an indicator represented on 1 bit. A configuration of the FLAGS register shows a synthetic overview of the execution of each instruction. For x86 the EFLAGS register has 32 bits but only 9 are actually used: CF, OF, DF, ZF, SF, AF, PF IF, TF.

The flags can be split into 2 categories:

→ with a previous effect generated by the Last Performed Operation (LPO): CF, PF, AF, ZF, SF, OF

→ having a future effect after their setting by the programmer to influence the way the next instructions are run: CF, TF, DF, IF ④

Carry flag is the transport flag. It will be set to 1 if in the LPO there was a transport digit outside the representation domain of the obtained result and set to 0 otherwise. Flags the unsigned overflow.

```
Ex: mov al, 150
    mov bl, 150 ; 150+150 = 300 ∉ [0,255] - which is the
    add al, bl    domain representable on a byte =>
                  => CF = 1.
```

Overflow flag marks the signed overflow. If the result of the LPO didn't fit the reserved space, then OF will be set to 1 and it will be set to 0 otherwise.

```
Ex: mov al, 100   the admissible domain on a byte (signed
    mov bl, 156   [-128, 127] =>
    sub al, bl    => 100 - (156) = 100 - (-256 + 156) = 100 - (-100)
                  = 200 ∉ [-128, 127] => doesn't fit on a byte
                  => OF = 1.
```

For multiplication, if we have the cases $b*b=b$; $w*w=w$ or $d*d=d$, then "no multiplication overflow" => CF=OF=0.
For $b*b=w$ or $w*w=d$ or $d*d=$ quad -> CF=OF=1.
For division, the "division overflow" is signaled by a 'Run-time error' instead of setting the flags.

Zero Flag is set to 1 if the result of the LPO was zero and set to 0 otherwise (except for mul or div).

Ex:
```
mov al, 15
mov bl, 241
add al, bl
```
$241+15=256 \notin [0,255] \Rightarrow 256-256=0 \Rightarrow ZF=1.$

Sign Flag is set to 1 if the result of the LPO is strictly negative and is set to 0 otherwise.

Ex:
```
mov al, -1
mov bl, -1
add al, bl
```
$(-1)+(-1)=-2 \in [-128,127] < 0 \Rightarrow SF=1.$

Parity flag - its value is set so that together with the bits 1 from the least significant byte of the representation of the LPO's result an odd number of digits to be obtained

Trap flag - is a debugging flag. If set to 1, then the machine stops after every instruction

Interrupt flag - if set to 1 interrupts are allowed, if set to 0 interrupts will not be handled.

Direction flag - if set to 0, the parsing of a sting will be performed in ascending order and in descending order if set to 1.

Auxiliary flag - shows the transport value from bit 3 to bit 4 of the LPO's result.

For setting the flags at ① we have the following instructions,

Carry flag:

CLC - clear carry flag => CF=0

STC - set carry flag => CF=1

CMC - complement carry flag => if CF=0 then CF=1
if CF=1 then CF=0.

Direction flag:

CLD - clear direction flag => DF=0

STD - set direction flag => DF=1

Interrupt flag:

CLI - clear interrupt flag => IF=0 (on 16 bits programming

STI - set interrupt flag -> IF=1.

- there are no instructions to directly access the value of TF.

PUSHF transfers all the flags on top of the stack (the contents of EFLAGS). The values of the flags are not affected by this instruction. The POPF instruction extracts the word from the top of the stack and transfers its contents into the EFLAGS register.