Flag Register

Flag Register:

It is a status register which indicates the current state of the processor. Computer makes decision based on the current state of the processor. In 8086, a 16 – bit register is used as a flag register. The flags stored in the flag register can be classified as two types. They are i) status flag and ii) control flag.

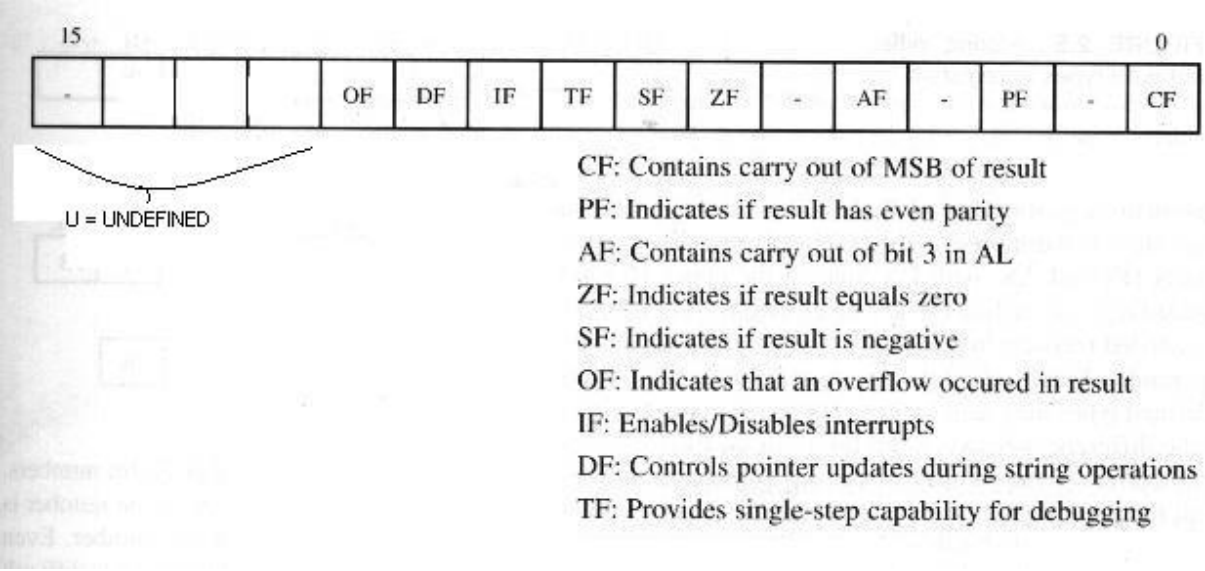
There are six status flags –

Carry flag (CF) Parity Flag (PF) Auxiliary Flag (AF) Zero Flag (ZF)

Sign Flag (SF) Overflow Flag (OF)

and three control flags –

Trap Flag (TF) Interrupt Flag (IF) Direction Flag (DF)



CF = 1; if there is a carry out form the msb on addition or borrow into the msb on subtraction.

0; otherwise.

PF = 1; if the low byte of a result has an even number of one.

1st 8 bits

0; otherwise.

AF = 1; if there is a carry out on addition or borrow into on subtraction in BCD operation.

0; otherwise.

ZF = 1; if a result is zero.

0; otherwise.

SF = 1; if msb of a result is 1.

0; if msb of a result is 0.

OF = 1; if signed overflow occurred.

0; otherwise.

Usually after executing an instruction, the flags are altered to reflect the result. But some instruction does not affect any flag, some affects all. The list below shows the instructions and their effects on the flags:

Instructions	Affect flags
MOV/ XCHG	none
ADD/ SUB	all
INC/DEC	all except CF
NEG	all (CF = 1 unless result is zero)

Example-1: ADD AX, BX where AX = FFFFh and BX = FFFFh. Show the changes of flags.

Solution:

FFFFh

\+ FFFFh

1 FFFEh

The result stored in AX is FFFEh = 1111 1111 1111 1110

SF = 1 because msb is 1.

PF = 0 because there are 7 one in the lower byte of the result.

ZF = 0 because result is not zero.

CF = 1 because there is a carry out of the msb.

OF = 0 because sign bit of the result is same as that of the operands.

Example2: ADD AL, BL where AL = 80h and BL = 80h.

Solution:

$$\begin{array}{r} 80h \\ \underline{+\ 80h} \\ 1\ 00h \end{array}$$

The result stored in AL is 00h

SF = 0 because msb is 0.

PF = 1 because all the bits in the result is 0.

ZF = 1 because result is zero.

CF = 1 because there is a carry out of the msb.

OF = 1 because the sign bit of the result is not same as that of the operands

Example3: SUB AX, BX where AX = 8000h and BX = 0001h.

Solution:

SUB AX, BX => CF=0
ADD AX, -BX => CF=1

$$\begin{array}{rl} 8000h & 1000\ 0000\ 0000\ 0000 \\ \underline{-\ 0001h} & 1111\ 1111\ 1111\ 1111 \\ 7FFFh & \end{array}$$

0000 0000 0000 0001 = 0001h
1111 1111 1111 1110 = ~0001h
+1

1111 1111 1111 1111 = -0001h

The result stored in AX is 7FFFh = 0111 1111 1111 1111.

SF = 0 because msb is 0.

PF = 1 because there are 8 one in the lower byte of the result.

ZF = 0 because result is not zero.

CF = 0 because there is no borrow into the msb.

OF = 1 because the result is positive.

Example 4: MOV AX, 5

Solution: The result stored in AX is 5.

None of the flags will be affected by MOV.

Example-5: NEG AX where AX = 8000h.

Solution:

8000h = 1000 0000 0000 0000

Two's complement = 1000 0000 0000 0000

The result stored in AX is 8000h.

SF = 1 because msb is 1.

PF = 1 because all the bits in the lower byte of the result is 0.

ZF = 0 because result is not zero.

CF = 1 because for NEG CF is always 1 unless the result is zero.

OF = 1 because the result is 8000h.

result= 80h or 8000h, OF=1

number of 1= 0--→ even