Flag Registers

- Circuits in CPU can take different decisions based on current state of CPU:
  - State Indicated by Flags
- Flags placed in Flag registers:
  - Status Flags:
    - Result of any computation
    - Located in bits: 0,2,4,6,7,11
  - Control Flags:
    - Enable or disable the operations of CPU
    - Located in bits: 8,9,10

Flag Resisters

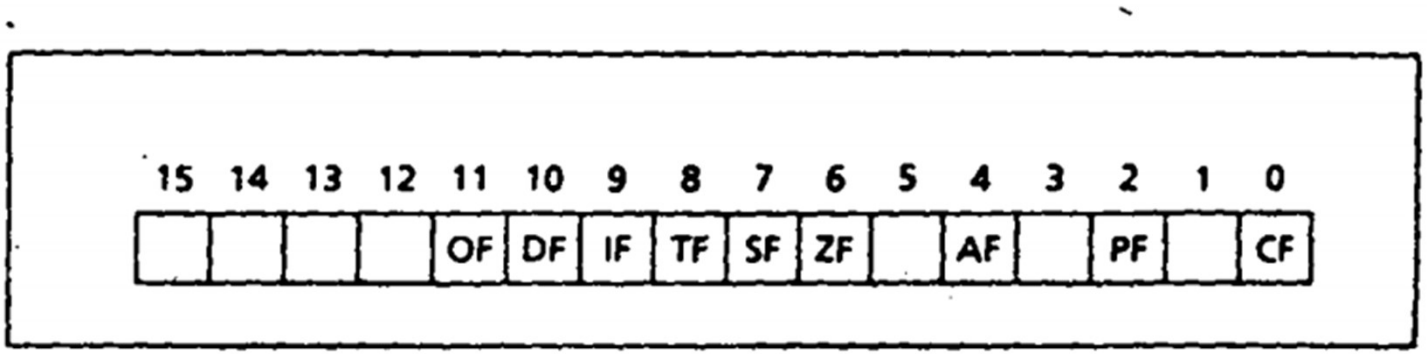

Table 5.1 Flag Names and Symbols

Status Flags

Bit	*Name*	*Symbol*
0	Carry flag	CF
2	Parity flag	PF
4	Auxiliary carry flag	AF
6	Zero flag	ZF
7	Sign flag	SF
11	Overflow flag	OF

Control Flags

Bit	*Name*	*Symbol*
8	Trap flag	TF
9	Interrupt flag	IF
10	Direction flag	DF

Status Flag

- Reflect the result of a computation
  - MOV AX,9
  - SUB AX,9
- When above instructions are executed result becomes zero
- Zero flag becomes 1
- Let’s discuss each status flag register

Carry Flag

- If Addition on MSB of a number generates a carry
  - CF becomes 1
- Add 255 and 3 in binary

Parity Flag

- After performing binary addition:
  - Exp: 000010, PF=0, Because odd number of ones
  - Exp:001100, PF=1 Because even number of ones
  - Used in Data Communication to check correctness of a message

Auxiliary Flag

- MSB bit carry was handled by CF
- Auxiliary flag becomes one:
  - If there is a carry on 3rd bit of a binary number
    - Exp: ADD 1111,0001

Zero Flag

- When operation result becomes zero, ZF becomes 1
- When operation result becomes non-zero, ZF becomes 0

Signed Flag

- If the result of binary operation is negative:
  - SF=1
  - Exp: subtract 7 from 4= -3
- If the result of binary operation is positive:
  - SF=0
  - Exp: subtract 4 from 7=3

Overflow Flag

- Processor sets OF=1, if an overflow occurs

Overflow

- After performing an arithmetic operation:
  - No overflow
  - Signed Overflow
  - Unsigned Overflow

```
  1111 1111 1111 1111
+ 0000 0000 0000 0001
---------------------
1 0000 0000 0000 0000
```

Problem

- Convert 24 into 8 bit binary number and 240 into 8 bit binary Number
- Add the binary numbers to gather and comment on the answer

Trap Flag

- System uses it for debugging:
  - 1 when single step mode(debugging) is needed to find errors
  - 0 when single step mode(debugging) is not required)

Interrupt Flag

- It becomes one when an Interrupt is called
- Zero when no Interrupt is called
- Exp: int21h

Direction flag

- Controls direction
- Exp: Reverse string
- Decrements offset when 1
- Does not decrement offset when zero

How Instructions affect flags?

Instruction	*Affects flags*
MOV/XCHG	none
ADD/SUB	all
INC/DEC	all except CF
NEG	all (CF = 1 unless result is 0, OF = 1 if word operand is 8000h, or byte operand is 80h)

Example 5.2 ADD AL,BL, where AL contains 80h, BL contains 80h.

Solution:

$$\begin{array}{r} 80h \\ +\ 80h \\ \hline 100h \end{array}$$

The result stored in AL is 00h.

SF = 0 because the msb is 0.

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

ZF = 1 because the result is 0.

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

OF = 1 because the numbers being added are both negative, but the result is 0 (as a binary addition, there is no carry into the msb but there is a carry out).

Solve

Example 5.4 INC AL, where AL contains FFh.

Solution

The result stored in AL is 00h. SF = 0, PF = 1, ZF = 1. Even though there is a carry out, CF is unaffected by INC. This means that if CF = 0 before the execution of the instruction, CF will still be 0 afterward.

OF = 0 because numbers of unlike sign are being added (there is a carry into the msb and also a carry out).