Microprocessor and Assembly Language CSC-321

Sheeza Zaheer

Lecturer

COMSATS UNIVERSITY ISLAMABAD

LAHORE CAMPUS

The Processor Status and Flag Register

OUTLINE

- **The Processor Status and Flag Register**
  - Introduction
  - Status Flags
  - Examples
  - In emu8086
- **References**
  - **Chapter 5,** Ytha Yu and Charles Marut, “Assembly Language Programming and Organization of IBM PC

The FLAG Register

- Nine individual bits called as **flag** are used to represent the 8086 processor state.
- Flags are placed in **FLAG Register**.
- Two types of flags:
  - **Status Flags**: Reflects the result of a computation. Located in bits: 0, 2, 4, 6, 7 and 11.
  - **Control Flags**: Used to enable/disable certain operations of the processor. Located in bits 8, 9 and 10.

Flags

				Flags$_H$								Flags$_L$			
X	X	X	X	OF	DF	IF	TF	SF	ZF	X	AF	X	PF	X*	CF

*Bits marked X are undefined.

Carry flag

Overflow

Parity flag

Direction

Auxiliary flag

Interrupt enable

Trap

Zero

Sign

6 are status flags

3 are control flag

The Status Flags (Carry Flag)

1. **Carry Flag:**

- The Carry Flag is set to 1 when there is a carry out from MSB on addition or there is a borrow into the MSB on subtraction. Also affected by shift and rotate instructions.
- **Examples:**
- FFh + 11h = 110h (If a register can store only 1 byte, then where to store the carry generated by MSB?)

Parity Flag

2. **Parity Flag**:

- PE (Even Parity): If the low byte of a result has an even number of one bits. For Even parity, PF =1
- PO (Odd Parity): If the low byte of a result has odd number of one bits. For Odd parity, PF = 0
- **Examples:**
- 1000 0001b – 1000 0010b = 11111111b (Number of one's in result = 8, so PF = 1)
- FFFFh + FFFFh = 1FFFEh (Number of one's in result = 7, so PF = 0)

Auxiliary Carry Flag

3. **Auxiliary Carry Flag**:

- The Auxiliary Carry Flag is set to 1 if there is a carry out from bit 3 on addition, or a borrow into bit 3 on subtraction.
- **Examples:**
- 1000 0001b – 0000 0010b = 01111111b (Borrow from bit 4 to bit 3)

Zero Flag

4. **Zero Flag:**

- Zero Flag is set when the result is zero.
- Zero Flag is unset when result is non-zero.
- **Examples:**

  0FFh – 0FFh = 00h

Sign Flag

5. Sign Flag:

- Set when MSB of a result is 1; it means the result is negative (signed interpretation)
- Unset when MSB is 0 i.e. result is positive.
- **Examples:**

  0FFh – 0FFh = 00h (MSB = 0, SF = 0)

Overflow Flag

6. **Overflow Flag:**
   - Set if signed overflow occurred, otherwise it is 0.

- **Overflow:**
  - Range of numbers that can be represented in a computer is limited.
  - If the result of an operation falls outside the defined range, Overflow occurs, and the truncated result will be incorrect.
- Four possible outcomes of an arithmetic operation:
  - No Overflow
  - Only Signed Overflow
  - Only Unsigned Overflow
  - Both Signed and Unsigned Overflow

How the Processor indicates overflow?

- The Processor sets:
  - Overflow Flag = 1 for Signed Overflow
  - Carry Flag = 1 for Unsigned Overflow

How the Processor determines that overflow occurred?

- **Unsigned Overflow:**
  - Addition: Carry out from MSB
  - The correct Answer is largest than the biggest unsigned number FFFFh for a word and FFh for a byte.
  - Subtraction: Borrow into MSB
- **Signed Overflow:**
  - Addition: Same sign but sum has a different sign (e.g.: when you add two positive numbers and answer is negative)
  - In Addition of two numbers with different signs, overflow is impossible.
  - Subtraction: If result has different sign than expected.

How Instructions Affect the Flags

Instructions	Affects Flags
MOV/XCHG	None
ADD/SUB	All
INC/DEC	All except CF
NEG	All (Carry Flag = 1 unless result is 0, Overflow Flag = 1 if word operand is 8000h, or byte operand is 80h

Example 5.1

- ADD AX, BX, where AX contains FFFFh, BX contains FFFFh
- **Solution**:
  - Actual Result = 1FFFEh
  - Result stored in AX = FFFEh
  - Flags:
    - SF = 1 because the MSB is 1
    - PF = 0 because there are 7 (odd number) of 1 bits in the low byte of the result.
    - ZF = 0 because nonzero result
    - CF = 1 because there is a carry out of the MSB on addition
    - OF = 0 because the sign of the stored result is the same as that of the numbers being added (in binary addition, there is a carry into the MSB and carry out from MSB also)

Example 5.2

- ADD AL, BL where AL contains 80h, BL contains 80h
- **Solution:**
  - Actual Result = 100h
  - Result in AL = 00h
  - Flags:
    - SF = 0 because MSB is 0
    - PF = 1 because all bits in result are 0
    - ZF = 1 because result is 0
    - CF = 1 because there is a carry out from MSB
    - OF = 1 because the numbers being added are both negative but the MSB in result is 0 (in binary addition, there is a no carry into the MSB but there is carry out from MSB.

Example 5.3

- SUB AX, BX where AX contains 8000h and BX contains 0001 h
- **Solution:**
  - Actual Result = Result in AX = 7FFFh
  - Flags:
    - SF = 0 because MSB is 0
    - PF = 1, Parity is Even because there are 8 one bits in the low byte of the result
    - ZF = 0 because result is nonzero
    - CF = 0 because a smaller unsigned number is being subtracted from a larger one
    - OF =1 because we are subtracting a positive number from a negative number but the result is positive (wrong sign of result)

Example 5.4

- INC AL where AL contains FFh
- **Solution:**
  - Actual Result = 100h
  - Result stored in AL = 00h
  - Flags:
    - SF = 0
    - PF = 1
    - ZF = 1
    - CF = 0 because CF is unaffected by INC
    - OF = 0 because number of unlike sign are being added (there is a carry into the MSB and also carry out from the MSB)

Example 5.5

- MOV AX, -5
- **Solution:**
  - Result in AX = -5 = FFFBh
  - None of the flags are affected by MOV

Example 5.6

- NEG AX where AX contains 8000h
- **Solution:**
  - Result in AX = 8000h (2’s complement)
  - SF = 1
  - PF = 1, in low byte of result, number of 1 bits is 0.
  - ZF = 0
  - CF = 1 because for NEG, CF is always 1 unless the result is zero
  - OF = 1 because there is no sign change

Flags In EMU8086