NEUB CSE 321 Lecture 7: Flags and their uses

Flags in 8086

Lets recall the flag register studied in lecture 2. The flag register is a16 bit register with 9 active bits. The bits are as follows

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				O	D	I	T	S	Z		A		P		C

The status flags are located in bits 0, 2, 4, 6, 7, and 11 and the control flags are located in bit 8, 9, and 10. The other bits have no significance: it's not important to remember. The functions of each of the flag bits are summarized in the table below.

Bit Mnemonic	Bit Name	Reset State	Function
OF	Overflow Flag	0	If OF is set, an arithmetic overflow has occurred.
DF	Direction Flag	0	If DF is set, string instructions are processed high address to low address. If DF is clear, strings are processed low address to high address.
IF	Interrupt Enable Flag	0	If IF is set, the CPU recognizes maskable interrupt requests. If IF is clear, maskable interrupts are ignored.
TF	Trap Flag	0	If TF is set, the processor enters single-step mode.
SF	Sign Flag	0	If SF is set, the high-order bit of the result of an operation is 1, indicating it is negative.
ZF	Zero Flag	0	If ZF is set, the result of an operation is zero.
AF	Auxiliary Flag	0	If AF is set, there has been a carry from the low nibble to the high or a borrow from the high nibble to the low nibble of an 8-bit quantity. Used in BCD operations.
PF	Parity Flag	0	If PF is set, the result of an operation has even parity.
CF	Carry Flag	0	If CF is set, there has been a carry out of, or a borrow into, the high-order bit of the result of an instruction.

Overflow

In a mathematical operation there are 4 possible outcomes relevant to overflow

1. No overflow
2. Signed overflow only
3. Unsigned overflow only
4. Both Signed overflow and Unsigned overflow

As an example of unsigned overflow but not signed overflow, suppose AX contains FFFFh, BX contains 0001h, and ADD AX,BX is executed. The binary result is-

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

If we are giving an unsigned interpretation, the correct answer is 10000h= 65536, but this is out of range for a word operation. A 1 is carried out of the msb and the answer stored in AX, -0000h, is wrong, so unsigned overflow occurred. But the stored answer is correct as a signed number, for FFFFh = -1. 0001h = 1, and FFFFh + 0001h = -1 + 1 = 0, so signed overflow did not occur.

PREPARED BY
SHAHADAT HUSSAIN PARVEZ

As an example of unsigned overflow but not signed overflow, suppose AX and BX both contains 7FFFh, and ADD AX,BX is executed. The binary result is-

$$\begin{array}{r} 0111\ 1111\ 1111\ 1111 \\ +\ 0111\ 1111\ 1111\ 1111 \\ \hline 1111\ 1111\ 1111\ 1110 = \text{FFFEh} \end{array}$$

The signed and unsigned decimal interpretation of 7FFFh is 32767. Thus for both signed and unsigned addition, 7FFFh + 7FFFh = 32767 + 32767 = 65534. This is out of range for signed numbers; the signed interpretation of the stored answer FFFEh is -2, so signed overflow occurred. However, the unsigned interpretation of FFFEh is 65534, which is the right answer, so there is no unsigned overflow.

How the processor indicates Overflow

OF= 1 : Signed overflow

CF = 1 : Unsigned Overflow

How the Processor Determines that Overflow Occurred

Unsigned Overflow

Operation	Overflow occurs when
Addition	there is carry
Subtraction	There is borrow

Signed overflow

- On addition of numbers with the same slgn, signed overflow occurs when the sum has a different sign.
- In addition of numbers With different signs, overflow Is impossible
- Subtraction of numbers with different signs is like adding numbers of the same sign. Signed overflow occurs if the result has a different sign than expected

Actually, the processor uses the following method to set the OF: If the carries into and out of the msb don't match- that is, there is a carry into the msb but no carry out, or if there is a carry out but no carry in - then signed overflow has occurred, and OF is set to 1. S

The affect of overflow on flags is summarized in the table below

Overflow type	Flags
No overflow	OF=0 &CF=0
Signed overflow	OF=1
Unsigned overflow	CF=1

How Instructions Affect the Flags

- MOV and XCHG - no flags are changed
- ADD and SUB - all flags affected
- INC and DEC - all except CF
- NEG - all flags affected
    - CF=0 only if value is 0
    - OF=1 only if value is -MAXINT
        - 80h or 8000h

PREPARED BY
SHAHADAT HUSSAIN PARVEZ

Examples

Example 5.1 ADD AX,BX, where AX contains FFFFh, BX contains FFFFh.

Solution:

```
  FFFFh
+ FFFFh
-------
1 FFFEh
```

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

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 the result is nonzero.

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 (as a binary addition, there is a carry into the msb and also a carry out).

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

Solution:

```
  80h
+ 80h
-----
100h
```

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).

Example 5.3 SUB AX,BX, where AX contains 8000h and BX contains 0001h.

Solution:

```
  8000h
- 0001h
-------
 7FFFh = 0111 1111 1111 1111
```

The result stored in AX is 7FFFh.

SF = 0 because the msb is 0.

PF = 1 because there are 8 (even number) one bits in the low byte of the result.

ZF = 0 because the result is nonzero.

CF = 0 because a smaller unsigned number is being subtracted from a larger one.

Now for OF. In a signed sense, we are subtracting a positive number from a negative one, which is like adding two negatives. Because the result is positive (the wrong sign), OF = 1.

PREPARED BY
SHAHADAT HUSSAIN PARVEZ

Example 5.4 INC AL, where AL contains FFh.

Solution:

```
  FFh
+ 1h
-----
 100h
```

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).

Example 5.5 MOV AX, -5

Solution: The result stored in AX is -5 = FFFBh.

None of the flags are affected by MOV.

Example 5.6 NEG AX, where AX contains 8000h.

Solution:

```
                    8000h = 1000 0000 0000 0000
        one's complement = 0111 1111 1111 1111
                                             +1
                           -------------------
                           1000 0000 0000 0000 = 8000h
```

The result stored in AX is 8000h.

SF = 1, PF = 1, ZF = 0.

CF = 1, because for NEG CF is always 1 unless the result is 0.

OF = 1, because the result is 8000h; when a number is negated, we would expect a sign change, but because 8000h is its own two's complement, there is no sign change.

In the next section, we introduce a program that lets us see the actual setting of the flags.

1. Marut Chapter 5 exercise question 1
2. Marut Chapter 5 exercise question 2
3. Marut Chapter 5 exercise question 3
4. Marut Chapter 5 exercise question 4

PREPARED BY
SHAHADAT HUSSAIN PARVEZ