Experiment Number : (11)

Signed Conditional Jump Instructions

Object:

To recognize the different between the signed and unsigned conditional jump instructions, and also where are used.

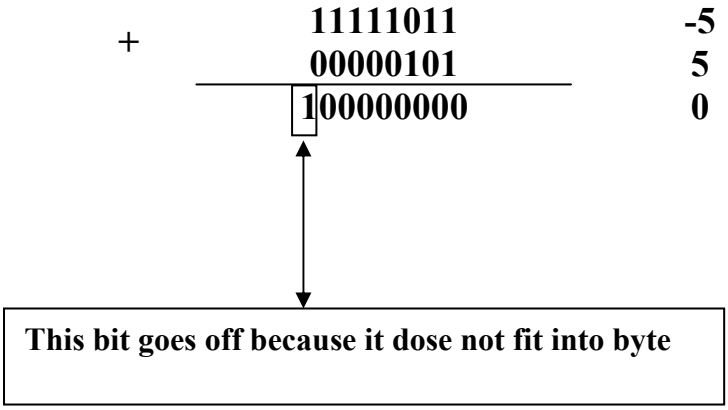
Theory:

Sign Number:

There is no way to say for sure whether the hexadecimal byte (0FFh) is positive or negative, it can represent both decimal value "255" & "-1".
8 bit can be used to create 256 combinations (include zero), so we simply presume the first 128 combinations (0..127) will represent positive number and next 128 combinations (128..255) will represent negative numbers.

In order to get "-5" we should subtract 5 from the number of combination (256), so will get 256 – 5 = 251.
Using this complex way to represent negative numbers has some meaning in math when you add "-5" to "5" you should get "zero". This is what happens when processor add tow bytes 5 and 251, the result gets over 255, because of the overflow processor gets zero.

Example:



When combinations 128..255 are used the high bit is always 1, so this maybe used to determine the sign of the number.

The same principle is used for words (16 bit values), 16 bits create 65536 combinations, first 32768 combinations (0..32767) are used to represent positive numbers, and next 32768 combinations (32768..65535) represent negative numbers.

Overflow:

Any changing of the number from positive +ve to negative –ve or changing from negative –ve to positive +ve is represent the over flow.

Example:

Let AL = 127 in decimal
BL = 1
AL = AL + BL = 128
The OF flag will set as one because 127 represent +ve and 128 represent –ve value (changing from positive to negative).

AL = 129 in decimal
BL = 2
AL = AL – BL = 127
The OF flag will set as one because 129 represent -ve and 127 represent +ve value (changing from negative to positive).

AX = 32767 in decimal
BX = 1
AX = AX + BX = 32768
The OF flag will set as one because 32767 represent +ve and 32768 represent –ve value (changing from positive to negative).

AX = 32770 in decimal
BX = 4
AX = AX – BX = 32766
The OF flag will set as one because 32770 represent -ve and 32766 represent +ve value (changing from negative to positive).

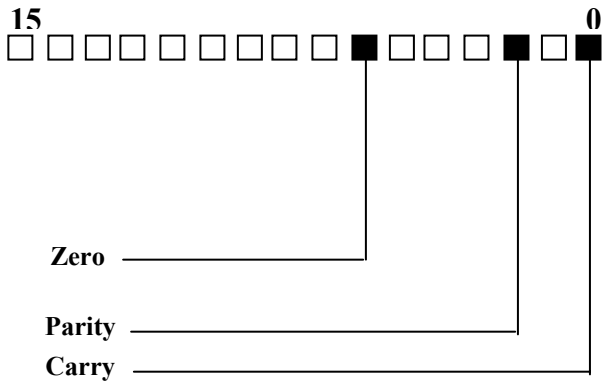
Jump instructions for signed number:

1.	**JE Label**	**Jump if equal**	**ZF = 1**
2.	**JZ Label**	**Jump if zero**	**ZF = 1**
3.	**JG Label**	**Jump if grater**	**ZF = 0 & SF=OF**
4.	**JL Label**	**Jump if less**	**SF ≠ OF**
5.	**JGE Label**	**Jump if grater or equal**	**SF=OF**
6.	**JLE Label**	**Jump if less or equal**	**ZF = 1 OR SF≠OF**

Some of the above instructions can be negative as seen below:

1.	**JNE Label**	**Jump if not-equal**	**ZF = 0**
2.	**JNZ Label**	**Jump if not-zero**	**ZF = 0**
3.	**JNG Label**	**Jump if not-grater**	**ZF = 1 OR SF≠OF**
4.	**JNL Label**	**Jump if not-less**	**SF = OF**

All of the above instructions test the some of the flag and its:



Example:

Write 8086 program to find the min value of DT_1, store the result into DT_2.
DT_1 = -5, 3, 1, -6, 2, 1, 0, -11, 10, 9

Solution:

```
.DATA
DT1 DB -5,3,1,-6,2,1,0,-11,10,10,9
DT2 DB 0
.CODE
MOV AX@ DATA
MOV DS, AX
MOV SI, OFFSET DT1
MOV DI, OFFSET DT2
MOV CX, 0009
MOV BX, 0
MOV AL, [SI+BX]
L: INC BX
CMP AL, [SI+BX]
JLE M
MOV AL. [SI+BX]
M: LOOP L
MOV [DI], AL
RET
```

Procedure:

You have 5 numbers (-5, 4, -7, 4, 0) write 8086 program to do:

1. **Let above program is DT_1.**
2. **Let DS=1000H , SI=0000 , DI=0050H**
3. **Find the no. of the number that is negative.**
4. **Store the results into DT_2.**
5. **Execute the above program and find the results.**

Home Work:

1. **Write 8086 program to find the max and min values of the DT_1, store the results into DT_2**

   DT_1 = -5, 3, 1, -6, 2, 1, 0, -11, 10, 9
2. **Write 8086 program to store the numbers (-5) to (5) into DT_1 without using the counter CX.**
3. **Write 8086 program to store the no. of the even numbers of the DT_1 store the results into DT_2.**

   DT_1 = -10, 10, 21, -1, 0, -11
4. **Write 8086 program to store the number (1) to DT_3 if the summation of DT_1 and DT_2 caused zero results otherwise store (0) in DT_3**

   DT_1 = 1, 5, 10, -12, 0

   DT_2 = 4, -5, -10, -12, 0