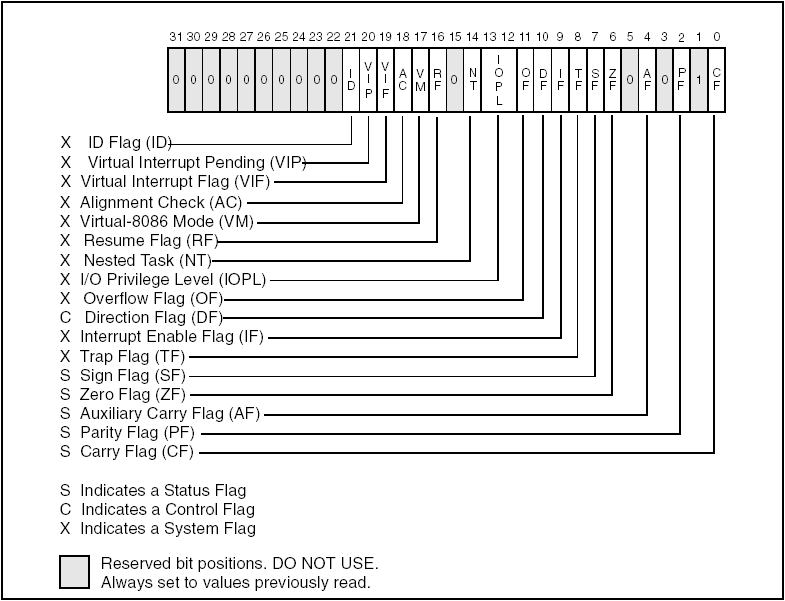
**Lab 03:**

**Arithmetic instructions affecting flag registers of Intel x-86 Architecture**

****

**Lab 03**

**Pre-lab:**

**Software required\*:**

* **MASM 6.15 (Assembler)**
* **32-bit operating systems(Windows)**
* **codesCh04.rar ( Irvine programs shared on group )**

*\*All these stuff are available on group as well in class*

**Goal:**

The purpose of lab 03 is to test flag register in register file. Upon completion, you will able to learn how these bit addressable flags are modified during execution of instruction.

**Target of Lab 03:**

1. Understanding various types of flags in register file of x86 Architecture.
2. To write specific code segments overwriting and manipulating register flags.

**Procedure:**

**Program Template:**

Before we start lab work a basic template for assembly language to be used later is provided for understanding

*TITLE Template for AL*

*INCLUDE Irvine32.inc*

***.data***

*; (insert variables here)*

***.code***

*main PROC*

*; (insert executable instructions here)*

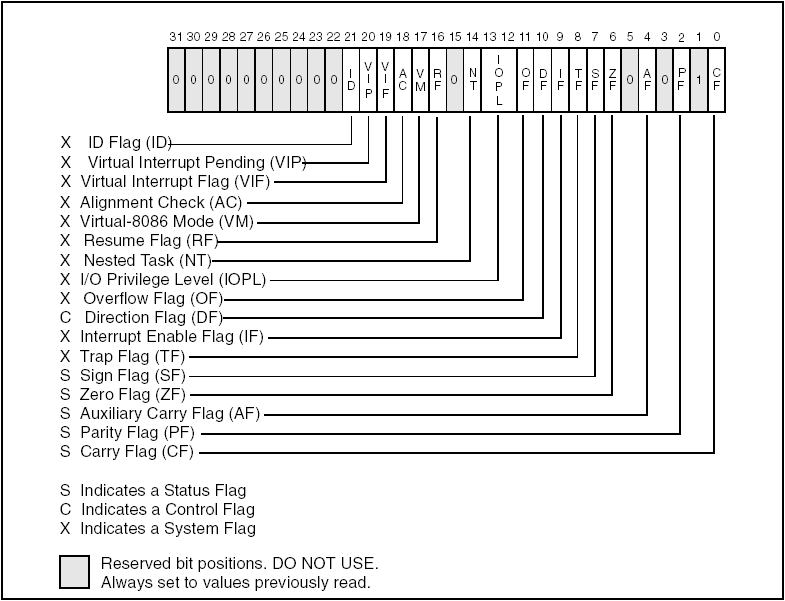
*exit*

*main ENDP*

*; (insert additional procedures here)*

*END main*

**Details of the FLAGS register**



*(Source:* ***IA-32 Intel® Architecture, Software Developer’s Manual****)*

|  |  |  |
| --- | --- | --- |
| **Bit** | **Name** | **Symbol** |
| 0 | Carry Flag | CF |
| 6 | Zero Flag | ZF |
| 7 | Sign Flag | SF |
| 11 | Overflow Flag | OF |

For this lab, we are only interested in bit numbers 0, 6, 7 and 11 containing flags tabulated below. Sample programs are written to see how each flag will be affected by various instructions performed on explicit data values.

|  |  |
| --- | --- |
| **Bit 6, Zero Flag (ZF)**  ;--------SAMPLE 1--------  mov cx, 1  sub cx, 1 ;cx=0, ZF=1  **Bit 7, Sign Flag (SF)**  ;--------SAMPLE 2--------  mov bx, 0  sub bx, 1 ; cx=-1, SF=1  add bx, 2 ; cx=1 , SF=0  **Bit 0, Carry Flag (CF)**  For unsigned arithmetic. Used to detect overflow in unsigned arithmetic.  ;--------SAMPLE 3--------  mov al, 0FFh  add al, 1 ; al=0, CF=1 | ;--------SAMPLE 4-------  mov ax, 00FFh  add ax, 1 ; ax=0100h, CF=0  **Food for thought:(why no carry flag asserted in this approach)**  **Bit 11, Overflow (OF)**  For signed arithmetic  ;--------SAMPLE 5-------  mov al, 127 ; i.e. +124  add al, 1 ; OF=1  ;--------SAMPLE 6-------  mov al, -128  sub a1, 1 ; OF=1 |

**Lab tasks:**

1-Your job is to write a code to test all register flags one by one. As a hint, this code could easily be written using template Assembly language code and sample code fragment both given above in this lab. Try to observe the contents of register using ***call dumpregs*** before .code segment terminates.

**Exercise :** Using the code samples presented earlier, fill out the following table indicating the values of the concerned flags after assembling and executing sample codes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sample No. | CF (bit 1) | ZF (bit 6) | SF (bit 7) | OF (bit 11) |
| Sample 1 | 0 | 1 | 0 | 0 |
| Sample 2 |  |  |  |  |
| Sample 3 | 1 | 1 | 0 | 0 |
| Sample 4 | 0 | 0 | 0 | 0 |
| Sample 5 | 0 | 0 | 1 | 1 |
| Sample 6 | 0 | 0 | 0 | 1 |

2-Download **codesCh04.rar** file from group in Lab folder containing all the AL codes provided by Irvine. Unzip **to** open ***addsub3.asm*** program file in text pad editor. Comment relevant sections to see the effect of individual instructions affecting flags. Assemble and run the program using **tools menu>>External tools>Build 32-Bit MASM** and **tools menu>>External tools>Run ASM Prog, respectively**.

**Notes / Comments:**