**Computer Organization and Architecture (EET2211)**

**LAB VII: Swap the upper nibble of a word with the lower nibble content of an accumulator**

**Siksha ‘O’ Anusandhan Deemed to be University, Bhubaneswar**

|  |  |  |  |
| --- | --- | --- | --- |
| **Branch:** Computer Science and Engineering **Section:** ‘D’ | | | |
| **S. No.** | **Name** | **Registration No.** | **Signature** |
| 52 | Saswat Mohanty | 1941012407 | **D:\Pics and Sign\sign.jpg** |

**Marks: \_\_\_\_\_\_/10**

**Remarks:**

**Teacher’s Signature**

**I. OBJECTIVE:**

1. Write a program to swap the upper nibble of a word with the lower nibble content of an accumulator.

**II. PRE-LAB**

**For Obj. 1:**

1. **Swap the upper nibble of a word with the lower nibble content of an accumulator.**

[5000h] = 1234h

Output: 3412h

1. **Write the assembly code.**

|  |
| --- |
| **org 100h**  **mov ax,0000h**  **mov ds,ax**  **mov ax,[5000h]**  **mov cl,08h**  **rol ax,cl**  **mov [5002h],ax**  **hlt**  **ret** |

**III. LAB:**

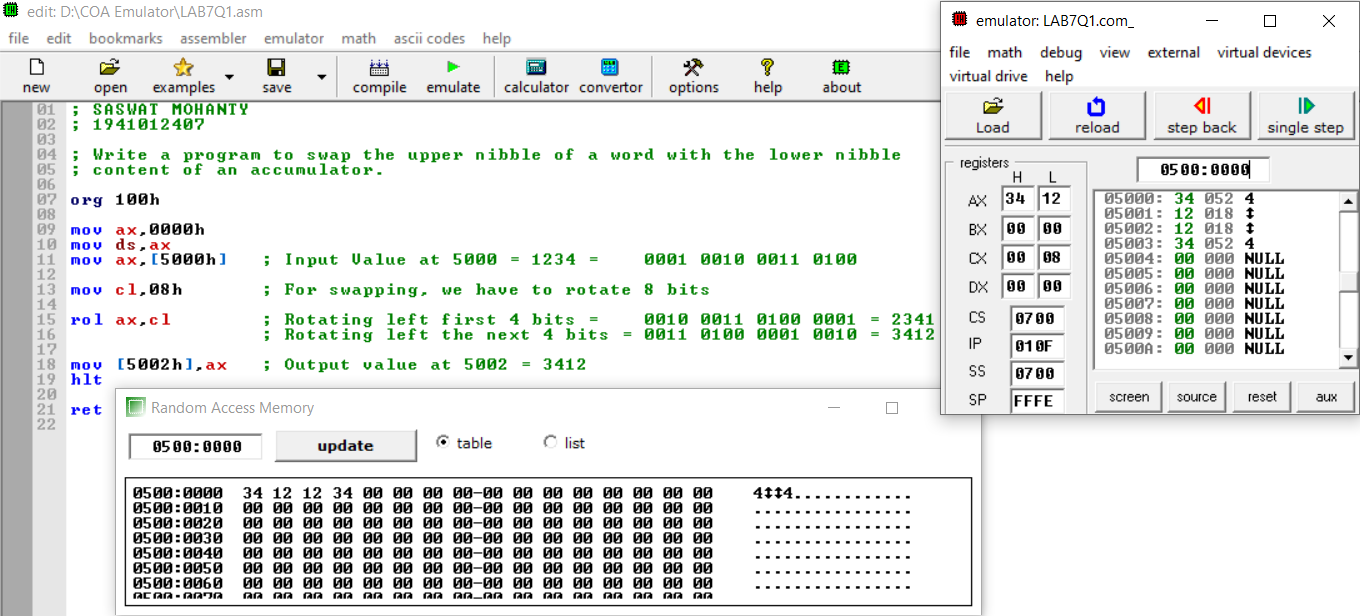
**Assembly Program:**

**For Obj. 1:**

|  |
| --- |
| **; SASWAT MOHANTY**  **; 1941012407**  **; Write a program to swap the upper nibble of a word with the lower nibble**  **; content of an accumulator.**  **org 100h**  **mov ax,0000h**  **mov ds,ax**  **mov ax,[5000h] ; Input Value at 5000 = 1234 = 0001 0010 0011 0100**  **mov cl,08h ; For swapping, we have to rotate 8 bits**  **rol ax,cl ; Rotating left first 4 bits = 0010 0011 0100 0001 = 2341**  **; Rotating left the next 4 bits = 0011 0100 0001 0010 = 3412**  **mov [5002h],ax ; Output value at 5002 = 3412**  **hlt**  **ret** |

**Observations (with screen shots):**

**For Obj. 1:**

****

**Conclusion:**

It can be concluded that swap the upper nibble of a word with the lower nibble content of an accumulator when dry run and executed in system found to be same. Thus, the program to swap the nibbles was executed.

**IV. POST LAB:**

**Explain briefly the advantages of memory segmentation in 8086.**

Advantages of memory segmentation in 8086:-

* It allows to processes to easily share data.
* It allows extending the address ability of the processor, i.e. segmentation allows the use of 16 bit registers to give an addressing capability of 1 Megabytes. Without segmentation, it would require 20 bit registers.

**Explain the IAS instruction format.**

The IAS machine was a binary computer with a 40-bit word, storing two 20-bit instructions in each word. The memory was 1,024 words (5.1 kilobytes). Negative numbers were represented in two's complement format. It had two general-purpose registers available: the Accumulator (AC) and Multiplier/Quotient (MQ).

**Briefly explain the following flags of 8086:**

1. **Carry Flag (CF) b) Parity Flag (PF) c) Adjust Flag (AF)**
2. **Zero Flag (ZF) e) Sign Flag (SF) f) Overflow Flag (OF)**
3. **Carry Flag (CF): -** Holds the carry after addition or borrow after subtraction. Also indicates some error conditions as dictated by some programs and procedures.
4. **Parity Flag (PF): -** PF=0= odd parity; PF=1=even parity
5. **Adjust Flag (AF): -** Holds the carry (half carry) after addition or borrow after subtraction between bit positions 3 and 4 of the result (e.g. in BCD addition or subtraction)
6. **Zero Flag (ZF): -** Shows the result of the arithmetic or logic operation.
7. **Sign Flag (SF): -** Holds the sign of the result after an arithmetic/logic instruction execution.
8. **Overflow Flag (OF): -** Overflow occurs when signed numbers are added or subtracted. An overflow indicates the result has exceeded the capacity of the machine.