Capital University of Science and Technology

Department of Computer Science

CS 2523 – Computer Organization and Assembly Language

QUIZ NO. 7: Bit Manipulation, Shift Instructions, Subroutine, Stack

CLO: 1. Define concepts in the design of microprocessor as state machine and designing its data path and its controller. [C1- Remembering]

CLO: 3. Implement assembly programs of intermediate complexity using the intel 8088 architecture. The student should also be able to convert intermediate complexity program in high level language into assembly code. [C3- Applying]

Semester: Summer 22 **Max Marks:** 10

Instructor: Ms. Tayyaba Zaheer **Date:** September 19, 2022 **Max Time:** 15 Minutes

Name: **Reg. No.**

Question No.1 [05 Marks]

Please choose the best possible option:

1. Provided with the following data, task is to invert Bit 2 and Bit 6 of a byte (remember that the bit on the right-hand side is Bit 0). The mask and the logical operator should be: (02 Marks)

7	6	5	4	3	2	1	0	**Bit Position**
1	0	0	0	0	1	0	0	**Data**

a) Mask=01000100 and Logical Operator=AND
b) Mask=01000100 and Logical Operator=OR
c) Mask=00100010 and Logical Operator=XOR
d) Mask=01000100 and Logical Operator=XOR

Solution: d

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**Bit Position**
1	0	0	0	0	1	0	0	**Data**
0	1	0	0	0	1	0	0	**Mask**
1	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**Logical Operator – XOR Result**

2. In Assembly, which of the following is valid Instruction Mnemonics? (01 Mark)
   a) ADD
   b) MOV
   c) INC
   d) All of the mentioned

Solution: d

3. In 16-bit multiplication, the multiplicand and the result should be kept in: (01 Mark)

a) 8-bit
b) 16-bit
c) 32-bit
d) 64-bit

Solution: c

4. Which register holds the address for a stack whose value is supposed to be directed at the topmost position? (01 Mark)
   a) Stack Pointer
   b) Stack Register
   c) Both a & b
   d) None of the mentioned

Solution: a

Question No. 2 [05 Marks]

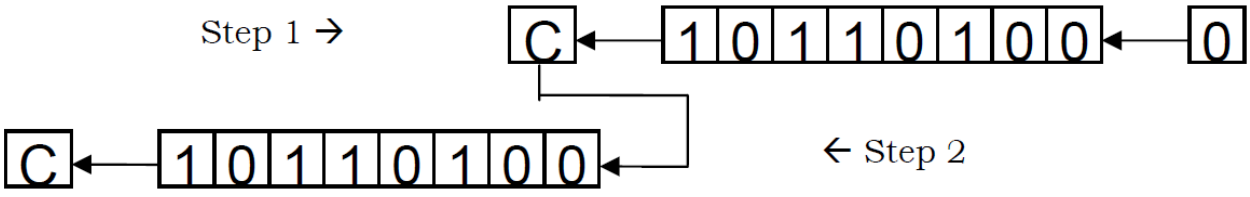
Consider two 32bit values placed in the memory. Write the code to declare the 32bit value in the memory and subtract them using the extended subtraction algorithm. Can we extend shift operations only up to 32 bits?

Solution:

```
dest:           dd    40000
src:            dd    80000

                mov   ax, [src]
                sub   word [dest], ax
                mov   ax, [src+2]
                sbb   word [dest+2], ax
```

Shift Instruction:




```
num1:           dd    40000

                shr   word [num1+2], 1
                rcr   word [num1], 1
```

- The same logic has worked. The shift placed the least significant bit of the upper half in the carry flag and it was pushed from right into the lower half.

- For a signed shift we would have used the shift arithmetic right instruction instead of the shift logical right instruction.
- **The extension we have done is not limited to 32 bits.**
- We can shift a number of any size say 1024 bits.
- The second instruction will be repeated a number of times and we can achieve the desired effect.
- Using two simple instructions we have increased the capability of the operations to effectively an unlimited number of bits.
- The actual limit is the available memory as even the segment limit can be catered with a little thought.