**Multiplication of two 8 bit nos. using Successive addition and Shift and add method**

**TITLE:** Multiplication of two 8 bit nos. using Successive addition and Shift and add method

**OBJECTIVES:**

1. Understand the implementation.
2. To interpret the Microprocessor Interfacing paradigms.
3. To express and apply the method of odd, add and shift method.
4. Understand implememtation of arithmetic instruction of 8086.

**PROBLEM STATEMENT:**

Write 8086/64 ALP to perform multiplication of two 8 bit hexadecimal nos. Use successive addition & shift & add method, Accept i/p from the user.

**HARDWARE REQUIRED:**

**CPU**: Intel i5 Processor

**OS**:Windows XP (16 bit execution), Fedora 18 32 & 64 bit execution

**SOFTWARE REQUIRED:**

**Editer**: gedit, GNU Editor

**Assembler**: NASM (Netwide Assembler)

**Linker**: GNU Linker

**INPUT:** Two hex nos.

For e.g. AL=12H, BL= 10H

**OUTPUT:**

Result : D120H

**THEORY:**

There are 5 basic form of define reverse directives.

**Directives** **Purpose**

DD Define byte

DW Define word

DD Define doubleword

DQ Define quad word

DT Define Ten byte

RESB Reserve byte

RESW Reserve word

RESQ Reserve quad word

REST Reserve ten word

**Instructions Needed:**

MOV : Move or copy word

ROR : Rotate to right

AND : Logical AND

INC : Increment

DEC : Decrement

JNZ : Jump if not zero

CMP : Compare

JNC : Jump if no carry

JBE : Jump if below

**Shift & Add method:**

The method taught in school for multiplying decimal no. is based on calculated partial products, shifting it to the left & then adding them together. Shift & add multilplication is similar to the multiplication performed by paper & pencil. This method adds the multiplicand X to itself Y times where Y denotes the multiplier. To multiply two nos. by paper & pencil placing the intermediate product in the appropriate positions to the left of earlier product.

1. Consider 1 byte is in AL & another in BL
2. We have to multiply byte in AL with byte in BL
3. In this method, you add 1 with itself & rotate other no. each times &shift it by 1 bit n left along with carry
4. If carry is present add 2 NOS.
5. Initialize count to n as we are scanning for n digit decrement counter each time, the bits are added

The result is stored in AX, display the result.

Eg., AH=11H, BL=10H, Count=n

**Step 1:**

**AX=11 + 11 = 22H**

Rotate BL by 1 bit to left along with carry 0001 0000

**B1=10H 0010 0000 (20)**

**Step 2:**

Decrement count =3

Check for carry, carry is not there So Add with itself

**AX=22+22=44H**

Rotate BL to left

**BL=0 0000 0000 (00)**

No carry

**Step 3:**

Decrement count=2

Add no. with itself

**AX=44+44=88H**

Rotate BL to left

**B2=0 (carry) 1000 0000 (80)**

**Step 4:**

Decrement count=0

Add no. with itself,

**AX=88+88=110H**

Rotate BL to left

**BL=0 (carry) 1000 0000 (80)**

**Step 5:**

Decrement count =0, carry is generated

Add Ax, BX

**0110+0000=0110H**

i.e.,

**11H+10H=0110H**

**ALGORITHM:**

1. **Successive Addition**
2. Start
3. Get the 1st no. from user
4. Get 2nd no. from user the no. will get as counter.
5. Initialize result=0
6. Add the 1st no. of itself as multi times
7. Decrement counter
8. Compare the counter with ‘0’
9. If count ≠0

Goto step 5

1. Else
2. Display the result
3. Stop

**Shift Addition Method**

1. Start
2. Get the 1st no. from user
3. Initialize count =0
4. No.1= no\*2
5. Get the 2nd no. from user
6. Shift multiplier to left along with carry
7. Check for carry, if present goto step 4
8. No. 1= no.1 + shifted no.2
9. Decrement counter
10. If not zero, goto step 6
11. Display result
12. Stop

**FLOWCHART:**

Successive Addition Method:

Get the 1st No.

Init 2nd no. as want

Init result = 0

Result = Result + 1st no.

Decrement counter

Is count =0?

NO

YES

**Display result**

1. **Add & shift Method**

Init product=00 count=4

Get the multiplication & multiplexer i. e. NO.1 & NO.2

No.1=NO.1+NO.2

NO.2=2\*NO.2

i.e. left shift by one bit

Is array =1

From NO2?

No.1=No.1+No.2

Count=count+1

Is count=0?

Display result

**CONCLUSION:**

From this program we have studied the multiplication of 8 bit nos. and in this we have studied and implemented the program of successive addition and shift & add method.

Output:

;[root@comppl208 nasm-2.10.07]# nasm -f elf64 multi26.asm

[root@comppl208 nasm-2.10.07]# ld -o multi26 multi26.o

[root@comppl208 nasm-2.10.07]# ./multi26

\*\*\*Multiplication by add & shift\*\*\*

Enter two digit number: 50

Enter two digit number: 02

Multiplication is: 00A0

;[root@comppl208 nasm-2.10.07]# nasm -f elf64 muladd26.asm

[root@comppl208 nasm-2.10.07]# ld -o muladd26 muladd26.o

[root@comppl208 nasm-2.10.07]# ./muladd26

\*\*\*Multiplication by successive addition\*\*\*

Enter two digit number: 05

Enter two digit number: 20

Multiplication is: 00A0