Experiments No:10

Title: HEX to BCD and BCD to HEX

Problem Statement: Write X86/64 ALP to perform multiplication of two 8-bit hexadecimal numbers. Use successive addition and add and shift method. (use of 64-bit registers is expected).

Objective: To learn and understand shift & rotate Instruction

Outcomes: On completion of this practical ,students will be able to

C218.3: Analyze and apply logic to demonstrate processor mode of operation

Hardware Requirement: NA

Software Requirement: Ubuntu ,NASM etc.

Theory Contents :

Multiplying unsigned numbers

Multiplying unsigned numbers in binary is quite easy. Recall that with 4 bit numbers we can represent numbers from 0 to 15. Multiplication can be performed done exactly as with decimal numbers, except that you have only two digits (0 and 1). The only number facts to remember are that 0*1=0, and 1*1=1 (this is the same as a logical "and").

Multiplication is different than addition in that multiplication of an n bit number by an m bit number results in an n+m bit number. Let's take a look at an example where n=m=4 and the result is 8 bits

Decimal	Binary
	1010
	x011 0
10	0000
x6	1010
60	1010
	+0000

In this case the result was 7 bit, which can be extended to 8 bits by adding a 0 at the left. When multiplying larger numbers, the result will be 8 bits, with the leftmost set to 1, as shown.

Decimal	Binary
13 x14 182	1101 x111 0 0000 1101 1101 +1101

As long as there are n+m bits for the result, there is no chance of overflow. For 2 four bit multiplicands, the largest possible product is 15*15=225, which can be represented in 8 bits.

Multiplying signed numbers

There are many methods to multiply 2's complement numbers. The easiest is to simply find the magnitude of the two multiplicands, multiply these together, and then use the original sign bits to determine the sign of the result. If the multiplicands had the same sign, the result must be positive, if they had different signs, the result is negative. Multiplication by zero is a special case (the result is always zero, with no sign bit).

Multiplication and division can be performed on signed or unsigned numbers. For unsigned numbers, MUL and DIV instructions are used, while for signed numbers IMUL and IDIV are used.

The format of the multiplication & division instruction does not specify the multiplicand as it is implicitly specified depending on the size of Source Below

Consider the example given below 6 × 13 = 78

Sequential addition from row to row

```
      0 1  1 0
    ×  1  1 0 1      Sum:
    ---------
      0 1 1 0
                     00000110
    0 0 0 0          00000110
  0 1 1 0            00011110
+ 0 1 1 0            01001110
-----------
0 1 0 0 1 1 1 0      Product
```

4.4.2 Sequential Shift/Add-Method

Method to avoid adder arrays

• shift register for partial product and multiplier with each cycle,

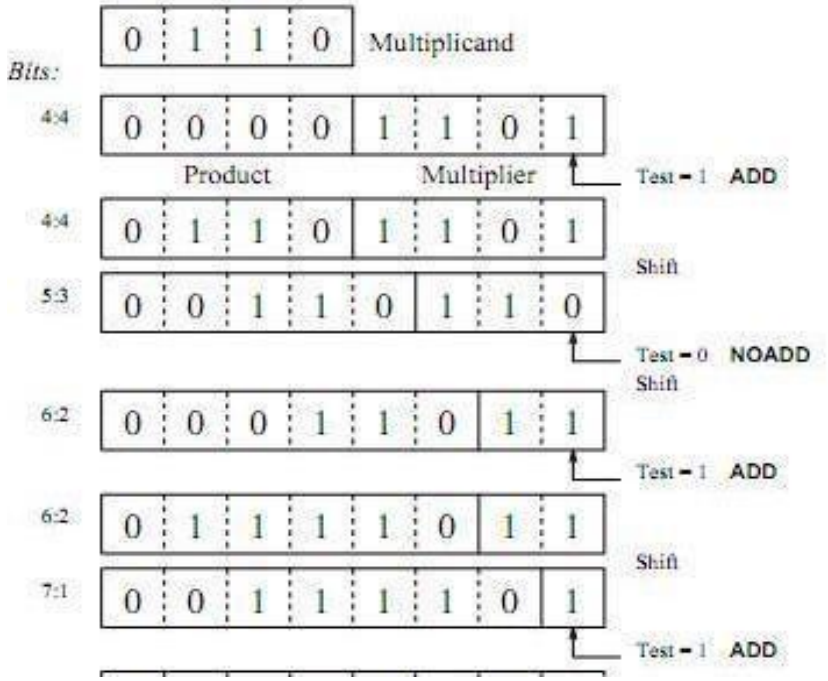
1. Partial product increases by one digit

2. Multiplier is reduced by one digit

• MSBs of partial product and multiplicand are aligned in each cycle

• not the multiplicand is shifted

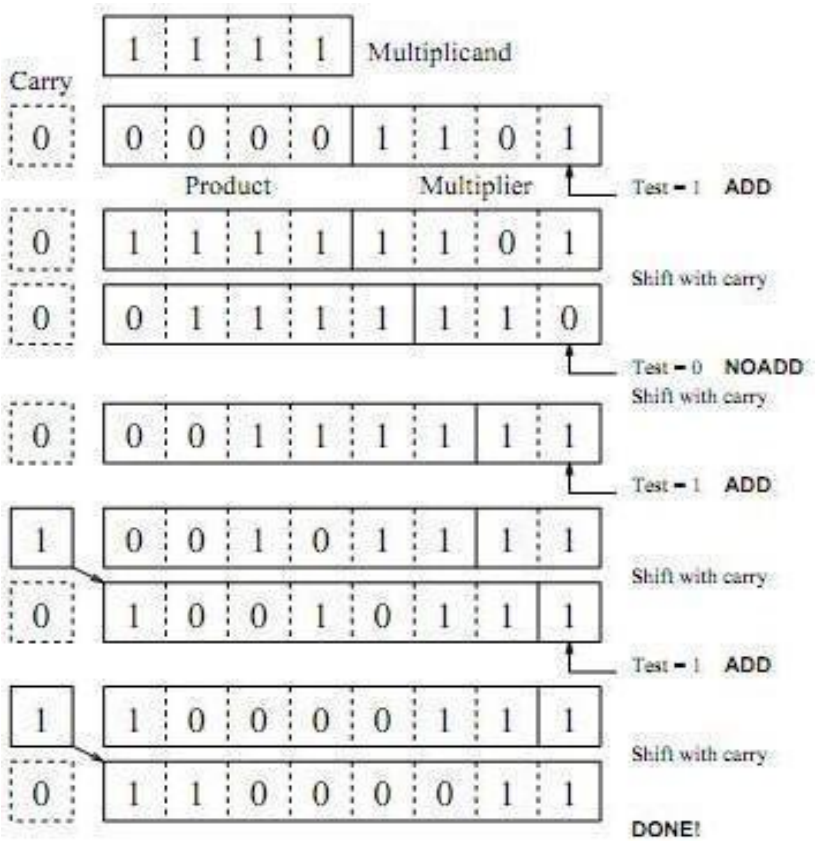
⇒ Partial product and multiplier are



Successive Addition Method

- Consider that a byte is for 8 bit Numbers present on Second byte is present in BL Register.
- We have to multiply the AL with the byte in BL
- Multiply the number using Successive Addition Method.
- In this method, one number is accepted and other number is taken as a counter.
- The first number is added with itself, till the counter decrements to zero. Result is stored in DX register, Display the result, using display routine

15 x 13 =195



4.1 Algorithm:

1. Start
2. Read multiplication& Multiplier for multiplication
3. Display menu
    a. Successive

Addition b.Add

& shift method

c.Exit

4. Read choice .if choice =a go to next step, if choice =

b go to Sep 9, if choice =c go to step 15

5. Assign sum =0 count= Multiplier
6. Sum =sum Multiplicand
7. Decrement count .if count >0 go to step 8
8. Print sum & go to step 3
9. Assign count =no of digit in multiplier, sum=0, shiftvar=0
10. Shift right Multiplier by 1
11. If carry flag set, sum =sum+ (Left shifted multiplicand by shiftvar)
12. Shitvar =shitvar+1
13. If count> 0 go to step 14
14. Print sum & go to step 3
15. Exit

4.2 Instructions needed:

1. MUL-Multiplication specified byte or word to word
2. SHR- Shift logical right byte or word, MSB to LSB and to CF

   SHL-Shift logical left byte or word, LSB to MSB and to CF
3. JMP-Unconditional jump to the specified location counter
4. JC-Jumps if carry is generated
5. JE/JZ-Jumps if equal or zero.

Directive Recommended:

1) MACRO- Start of MACRO statement
2) ENDM-end

4.3 Assignment Question:

1. Explain MACRO with example. Justify where macro is suitable than procedure.
2. Differentiate between PROC& MACRO
3. What is the difference between a rotate & a shift instruction?

Explain with an appropriate diagram.

4. Explain the difference between arithmetic shift & logical shift.
5. Describe execution of CALL instruction.
6. Solve 17 & 18eExample on Paper Manually for Both Methods using Program Logic ?

4.4 Oral Question:

1. Explain the Instruction used in the program
2. With example explain Add and Shift Multiplication
3. With example explain Successive Addition Multiplication
4. Suggest the alternative instructions for JE/JZ (Logic of program should not change)
5. Difference in multiplication by MUL and IMUL

Conclusion: In this way we studied shifting operation and multiplication using successive addition and add-shift method.

MPL Practical Oral Question Bank

Sr No	BL	Questions	Oral 1	Oral 2 (improvement)	Remark
1	2	Explain the Instruction used in the program?.			
2	2	Explain the difference between arithmetic shift & logical shift..			
3	1	With example explain Successive Addition Multiplication?			
4	2	Difference in multiplication by MUL and IMUL?			
5	2	Explain conversion of Ascii code to Hex code			
6	2	Explain conversion of Hex code to Ascii code ?			

Sign of Student