**Practical - 2**

**Aim:** Write an assembly program to perform Addition, Subtraction of :

1. two 8-bit numbers
2. two 16-bit numbers
3. two 32-bit numbers
4. two 64-bit numbers

Note : get the data from memory locations, result should be available at memory location

**Description of instructions used:**

**MOV:**

* The MOV instruction copies a word or byte of data from a specified source to a specified destination. The destination can be a register or a memory location. The source can be a register, a memory location or an immediate number. The source and destination cannot both be memory locations. They must both be of the same type (bytes or words). MOV instruction does not affect any flag.
* Syntax:

mov <reg>,<reg>

mov <reg>,<mem>

mov <mem>,<reg>

mov <reg>,<const>

mov <mem>,<const>

**ADD:**

* These instructions add a number from some source to a number in some destination and put the result in the specified destination.
* Syntax:

add <reg>,<reg>

add <reg>,<mem>

add <mem>,<reg>

add <reg>,<const>

add <mem>,<const>

**SUB:**

* These instructions subtract the number in some source from the number in some destination and put the result in the destination.
* Syntax:

sub <reg>,<reg>

sub <reg>,<mem>

sub <mem>,<reg>

sub <reg>,<const>

sub <mem>,<const>

**ADC:**

* Add with carry.Perform same operation as ADD, but add carry flag bit to the result.All conditional flags are affected by this
* Syntax:

add <reg>,<reg>

add <reg>,<mem>

add <mem>,<reg>

add <reg>,<const>

add <mem>,<const>

**SBB:**

* Subtract with borrow.Subtract Source operand and borrow flag (CF-reflect result of previous calculation) from destination operand and result is saved in destination operand
* Syntax:

sub <reg>,<reg>

sub <reg>,<mem>

sub <mem>,<reg>

sub <reg>,<const>

sub <mem>,<const>

**Code:**

1. **Two 8-bit numbers**

**Addition**

MOV CX,1010H

MOV DS,CX

MOV [0001H],56H

MOV [0003H],32H

MOV BL,[0001H]

MOV CL,[0003H]

ADD BL,CL

MOV CL,0H

MOV [0005H],BL

**Subtraction**

MOV CX,1010H

MOV DS,CX

MOV [0001H],56H

MOV [0003H],32H

MOV BL,[0001H]

MOV CL,[0003H]

SUB BL,CL

MOV CL,0H

MOV [0005H],BL

1. **Two 16-bit numbers**

**Addition**

MOV CX,1010H

MOV DS,CX

MOV [0001H],1234H

MOV [0003H],2143H

MOV BX,[0001H]

MOV CX,[0003H]

ADD BX,CX

MOV CX,0H

MOV [0005H],BX

**Subtraction**

MOV CX,1010H

MOV DS,CX

MOV [0001H],5678H

MOV [0003H],2143H

MOV BX,[0001H]

MOV CX,[0003H]

SUB BX,CX

MOV CX,0H

MOV [0005H],BX

1. **Two 32 bit numbers**

**Addition**

MOV CX,1010H

MOV DS,CX

MOV [0001H],1234H

MOV [0003H],2741H

MOV [0006H],1111H

MOV [0008H],2222H

MOV BX,[0003H]

MOV CX,[0008H]

ADD BX,CX

MOV CX,[0001H]

MOV AX,[0006H]

ADC CX,AX

MOV AX,0H

MOV [0011H],CX

MOV [0013H],BX

**Subtraction**

MOV CX,1010H

MOV DS,CX

MOV [0001H],5234H

MOV [0003H],2743H

MOV [0006H],1111H

MOV [0008H],2222H

MOV BX,[0003H]

MOV CX,[0008H]

SUB BX,CX

MOV CX,[0001H]

MOV AX,[0006H]

SBB CX,AX

MOV AX,0H

MOV [0011H],CX

MOV [0013H],BX

1. **Two 64 bit numbers**

**Addition**

MOV ax, 1010H

MOV ds, ax

MOV [1001H], 6541h

MOV [1003H], 4581h

MOV [1005H], 1236h

MOV [1007H], 5147h

MOV [1009H], 1111h

MOV [1011H], 2222h

MOV [1013H], 4444h

MOV [1015H], 6666h

MOV bx, [1007H]

MOV cx, [1015H]

ADD bx, cx

MOV cx, [1005H]

MOV ax, [1013H]

ADC cx, ax

MOV ax, [1003H]

MOV dx, [1011H]

ADC ax, dx

MOV dx, [1001H]

ADC dx, [1009H]

MOV [1019H], dx

MOV [1021H], ax

MOV [1023H], cx

MOV [1025H], bx

**Subtraction**

MOV ax, 1010H

MOV ds, ax

MOV [1001H], 6541h

MOV [1003H], 4581h

MOV [1005H], 1236h

MOV [1007H], 5147h

MOV [1009H], 1111h

MOV [1011H], 2222h

MOV [1013H], 4444h

MOV [1015H], 6666h

MOV bx, [1007H]

MOV cx, [1015H]

SUB bx, cx

MOV cx, [1005H]

MOV ax, [1013H]

SBB cx, ax

MOV ax, [1003H]

MOV dx, [1011H]

SBB ax, dx

MOV dx, [1001H]

SBB dx, [1009H]

MOV [1019H], dx

MOV [1021H], ax

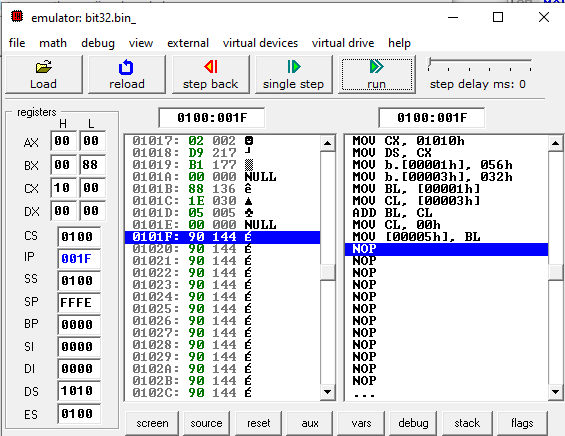
MOV [1023H], cx

MOV [1025H], bx

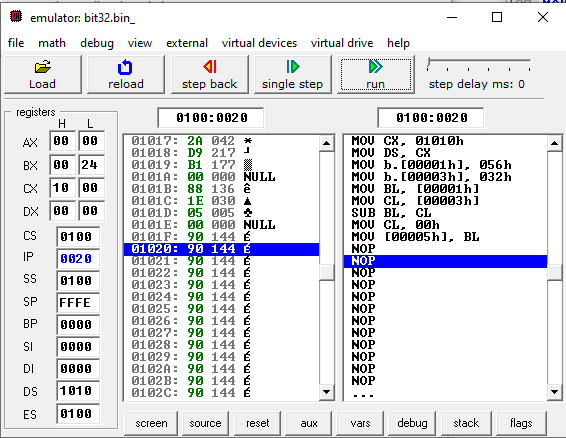
**Input/output:**

**Two 8-bit numbers**

**Addition**

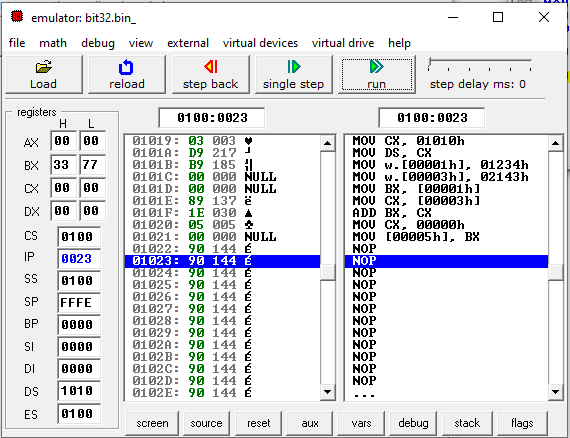


**Subtraction**

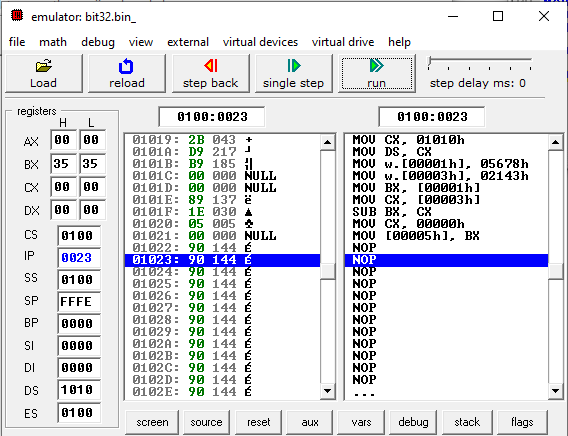


**Two 16-bit numbers**

**Addition**

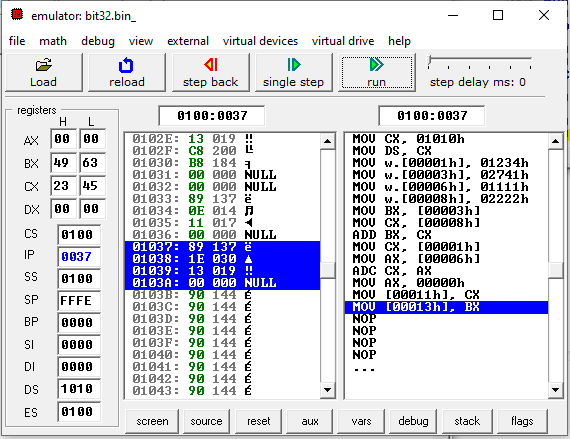


**Subtraction**

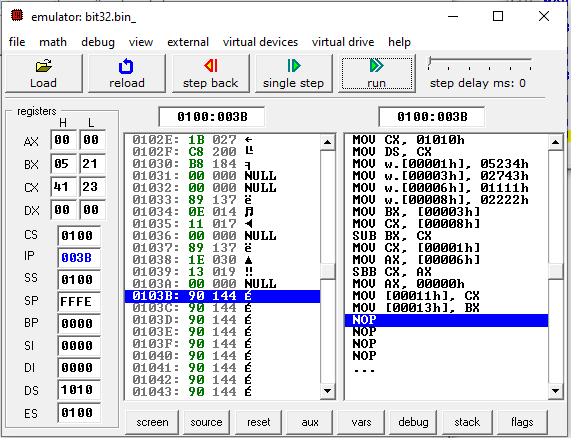


**Two 32 bit numbers**

**Addition**

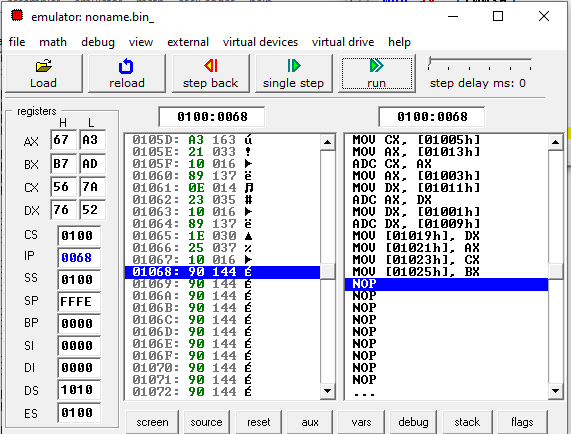


**Subtraction**

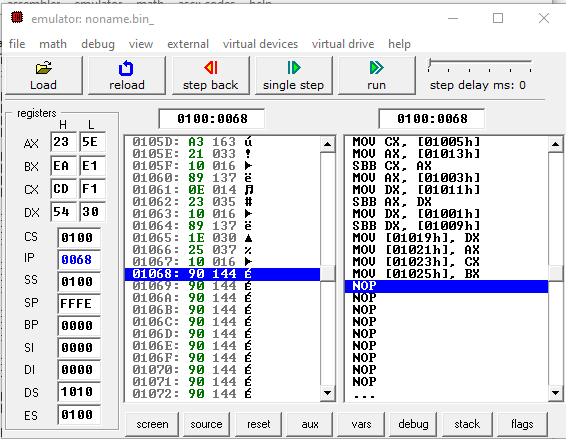
****

**Two 64 bit numbers**

**Addition**



**Subtraction**

****

**Conclusion**

In this practical we have learn and perforn practical about addtion and substraction of two 8,1,6,32,64 bit registers.