8086 Programs

- https://emu8086-microprocessor-emulator.en.softonic.com/

8 bit addition

```
MOV AL,[2000]
MOV BL,[2001]
ADD AL,BL
MOV[2002],AL
HLT
```

- **Output**

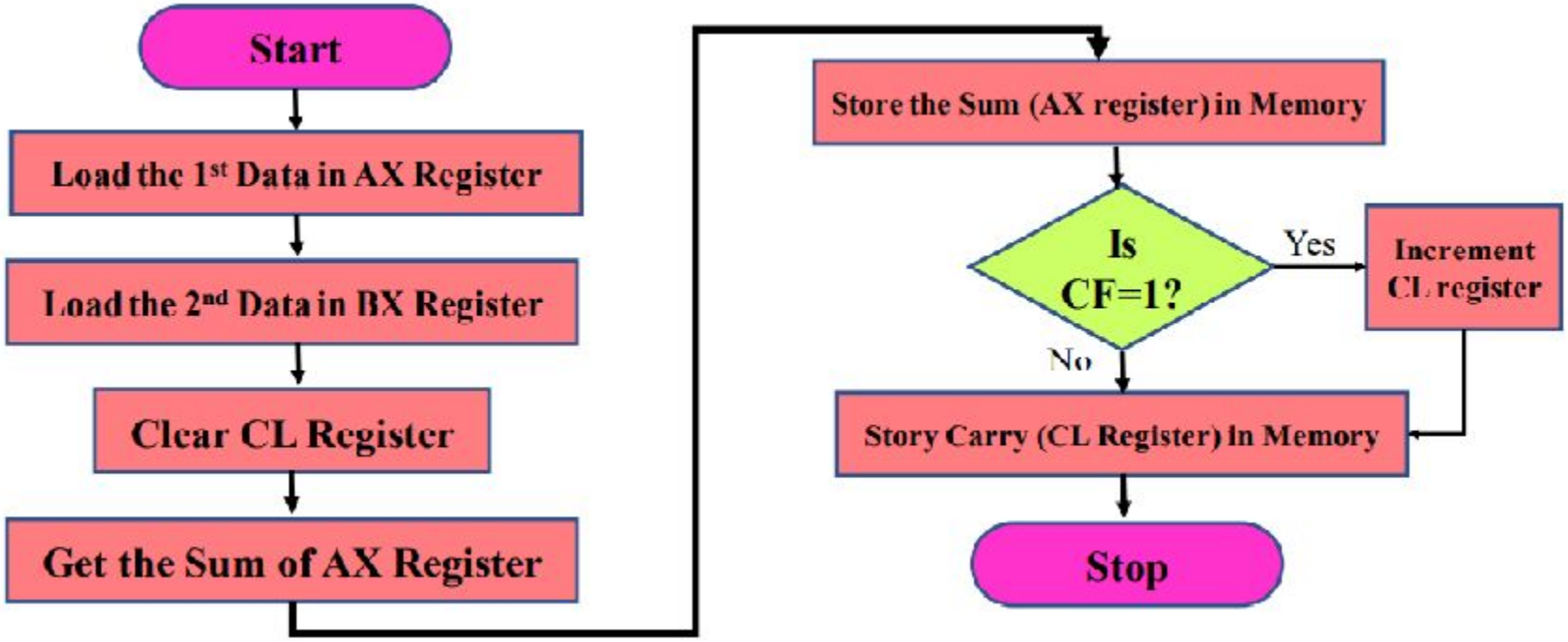
Memory location	content
2000	02
2001	04
2002	06

Write an Assembly Language Program to ADD two numbers of a 16 bit data.

- **Algorithm**

1. Load the first data in AX register
2. Load the second data in BX register
3. Clear CL register
4. Add the two data and get the sum in AX register
5. Store the sum in memory
6. Check for Carry, If Carry flag is set than go to next step, otherwise go to step 8.
7. Increment CL register
8. Store the Carry in Memory
9. Stop

Flow Chart


Start
Load the 1st Data in AX Register
Load the 2nd Data in BX Register
Clear CL Register
Get the Sum of AX Register
Store the Sum (AX register) in Memory
Is CF=1?
Yes
Increment CL register
No
Story Carry (CL Register) in Memory
Stop

16 bit Addition

```
MOV AX,[1000h]
 MOV BX,[1002h]
MOV CL,00h
ADD AX,BX
MOV [1004h],AX
 JNC jump
 INC CL
jump: MOV [1006h],CL
HLT
```

Example 1

AX	2316
BX	3243
Sum	5559

INPUT	
Memory Address	Content
1000	16
1001	23
1002	43
1003	32

AH	23	AL	16
BH	32	BL	43
CH		CL	00
DH		DL	

AH	55	AL	59
BH	32	BL	43
CH		CL	
DH		DL	

CF	0

OUTPUT	
Memory Address	Content
1004	59
1005	55
1006	00

Write an Assembly Language Program to Subtract two numbers of a 16 bit data.

```
MOV AX,[1000h]
MOV BX,[1002h]
MOV CL,00h
SUB AX,BX
JNC jump
INC CL
NOT AX
ADD AX,0001h
jump:
MOV [1004h],AX
MOV [1006h],CL
HLT
```

Multibyte Addition

Algorithm

1.Load the starting address of 1st data in SI register

2.Load the starting address of 2nd data in DI register

3.Load the starting address of result in BP register

4.Load the byte count in CL register

5.Let BX register be byte pointer. Initialize byte pointer as zero

6.Clear DL register to account for final carry

7.Clear Carry flag

8.Load a byte of 1st data in AL register

9.Add the corresponding byte of 2nd data in memory to AL register along with previous carry.

10.Store the sum in memory

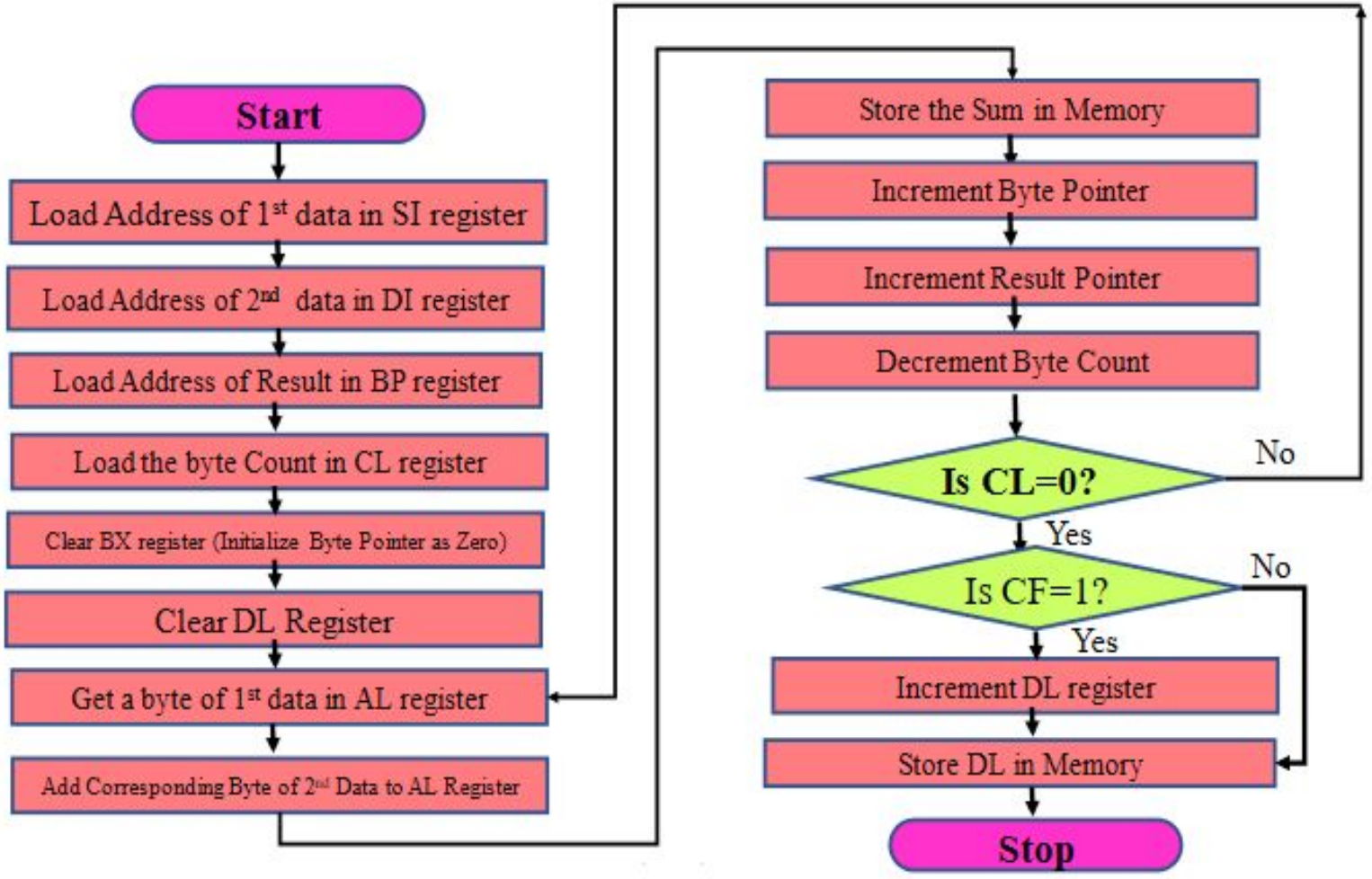
11.Increment the byte pointer (BX) and result pointer (BP).

12.Decrement the byte count (CL)

13.If byte count (CL) is zero, go to next step, otherwise go to step 8

14. Check for Carry. If carry flag is set then go to next step, otherwise for to step 16.
15. Increment DL register
16. Store final carry in memory
17. Stop

Flow Chart


Start
Load Address of 1st data in SI register
Load Address of 2nd data in DI register
Load Address of Result in BP register
Load the byte Count in CL register
Clear BX register (Initialize Byte Pointer as Zero)
Clear DL Register
Get a byte of 1st data in AL register
Add Corresponding Byte of 2nd Data to AL Register
Store the Sum in Memory
Increment Byte Pointer
Increment Result Pointer
Decrement Byte Count
Is CL=0?
No
Yes
Is CF=1?
No
Yes
Increment DL register
Store DL in Memory
Stop

Program

```
MOV SI,1000H
MOV DI,1011H
MOV BP,1021H
MOV CL,[SI]
INC SI
MOV BX,0000H
MOV DL,0000H
CLC
REPEAT:MOV AL,[SI+BX]
ADC AL,[DI+BX]
MOV [BP],AL
INC BX
INC BP
LOOP REPEAT
JNC JUMP
INC DL
JUMP:MOV [BP],DL
HLT
```

Example

INPUT	
Memory Address	Content
1000	05
1001	17
1002	72
1003	64
1004	C2
1005	F5

F 5 C 2 6 4 7 2 1 7

C 2 6 5 7 5 0 7 1 2

1 B 8 2 7 D 9 7 9 2 9

INPUT	
Memory Address	Content
1011	12
1012	07
1013	75
1014	65
1015	C2

Multiply two 16-bit nos

- MOV SI,1100H
- MOV AX,[SI]
- MOV BX,[SI+2]
- MUL BX
- MOV [SI+4],AX
- MOV [SI+6],DX
- HLT

Divide 32 bit data by 16 bit data

- MOV SI,1100H
- MOV AX,[SI]
- MOV DX,[SI+2]
- MOV BX,[SI+4]
- DIV BX
- MOV [SI+6],AX
- MOV [SI+8],DX
- HLT

Write an ALP to determine the sum of elements in an array

```
MOV CX,06h
MOV AX,0000h
MOV BX,0000h
MOV SI,1000h
REPEAT: MOV BL,[SI]
ADD AX,BX
INC SI
DEC CX
JNZ REPEAT
MOV DI,1011h
MOV [DI],AX
 HLT
```

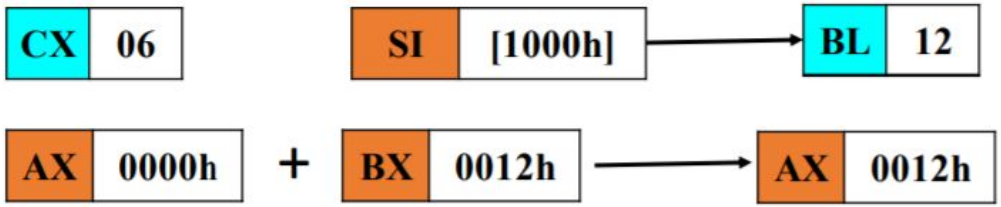
Example

CX	06

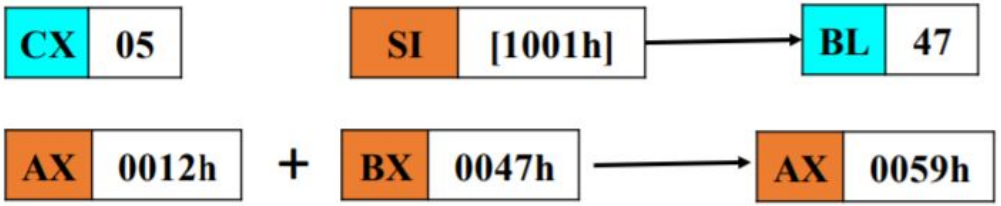
INPUT	
Memory Address	Content
1000	12
1001	47
1002	C2
1003	F5
1004	47
1005	56

OUTPUT	
Memory Address	Content
1011	AD
1012	02

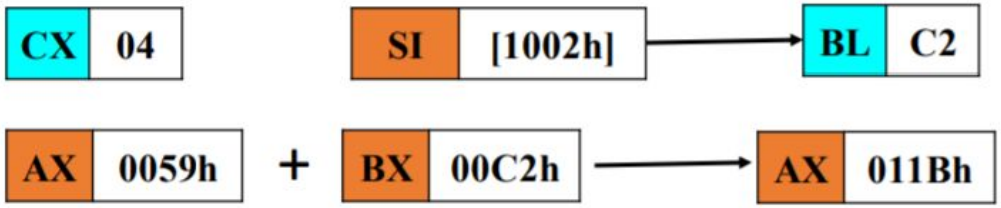
Iteration 1



Iteration 2



Iteration 3



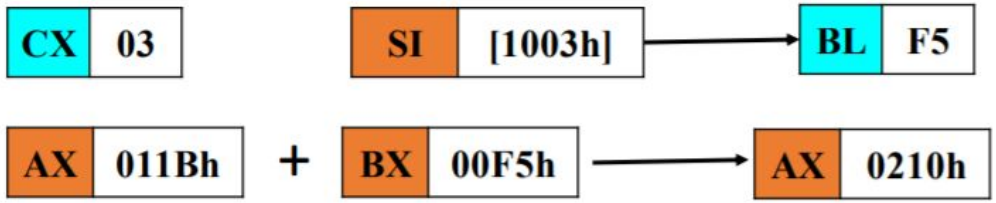
Example

CX	06

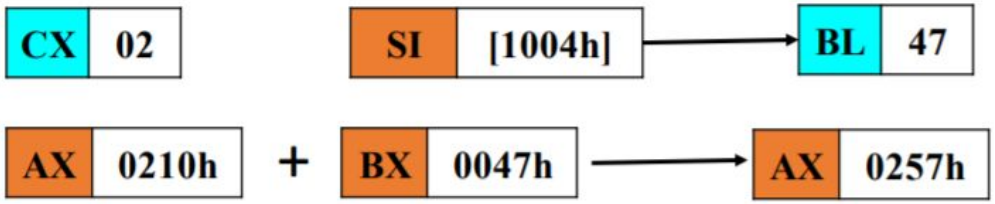
INPUT	
Memory Address	Content
1000	12
1001	47
1002	C2
1003	F5
1004	47
1005	56

OUTPUT	
Memory Address	Content
1011	AD
1012	02

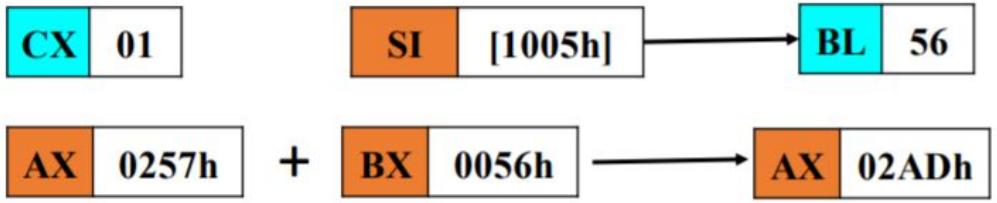
Iteration 4



Iteration 5



Iteration 6

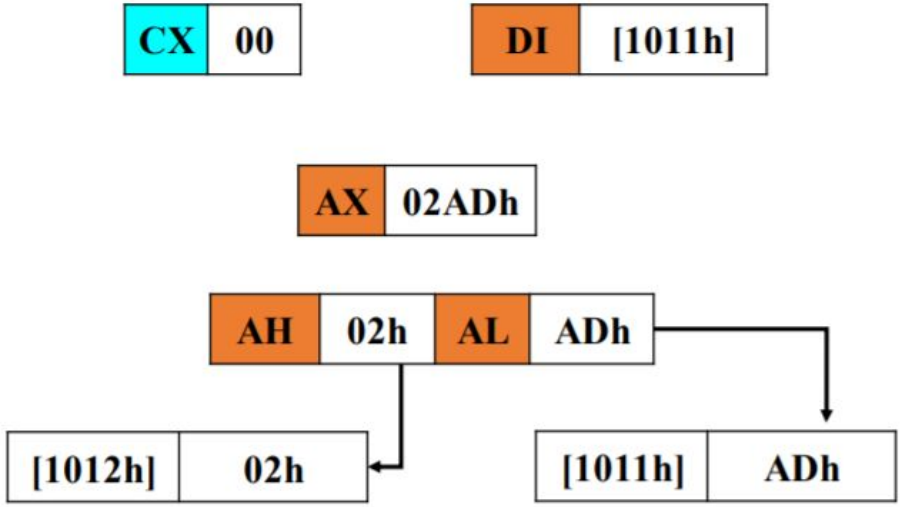


Example

CX	06

INPUT	
Memory Address	Content
1000	12
1001	47
1002	C2
1003	F5
1004	47
1005	56

OUTPUT	
Memory Address	Content
1011	AD
1012	02



Write an ALP to add two numbers of BCD data

Example

```
  4 5 7 8
  8 5 9 8
-----------
  C B 1 0
  6 6 6 6
-----------
1 3 1 7 6
```

```
0 1 0 0  0 1 0 1  0 1 1 1  1 0 0 0
1 0 0 0  0 1 0 1  1 0 0 1  1 0 0 0
-----------------------------------
1 1 0 1  1 0 1 1  0 0 0 1  0 0 0 0
0 1 1 0  0 1 1 0  0 1 1 0  0 1 1 0
-----------------------------------
1 0 0 1 1  0 0 0 1  0 1 1 1  0 1 1 0
```

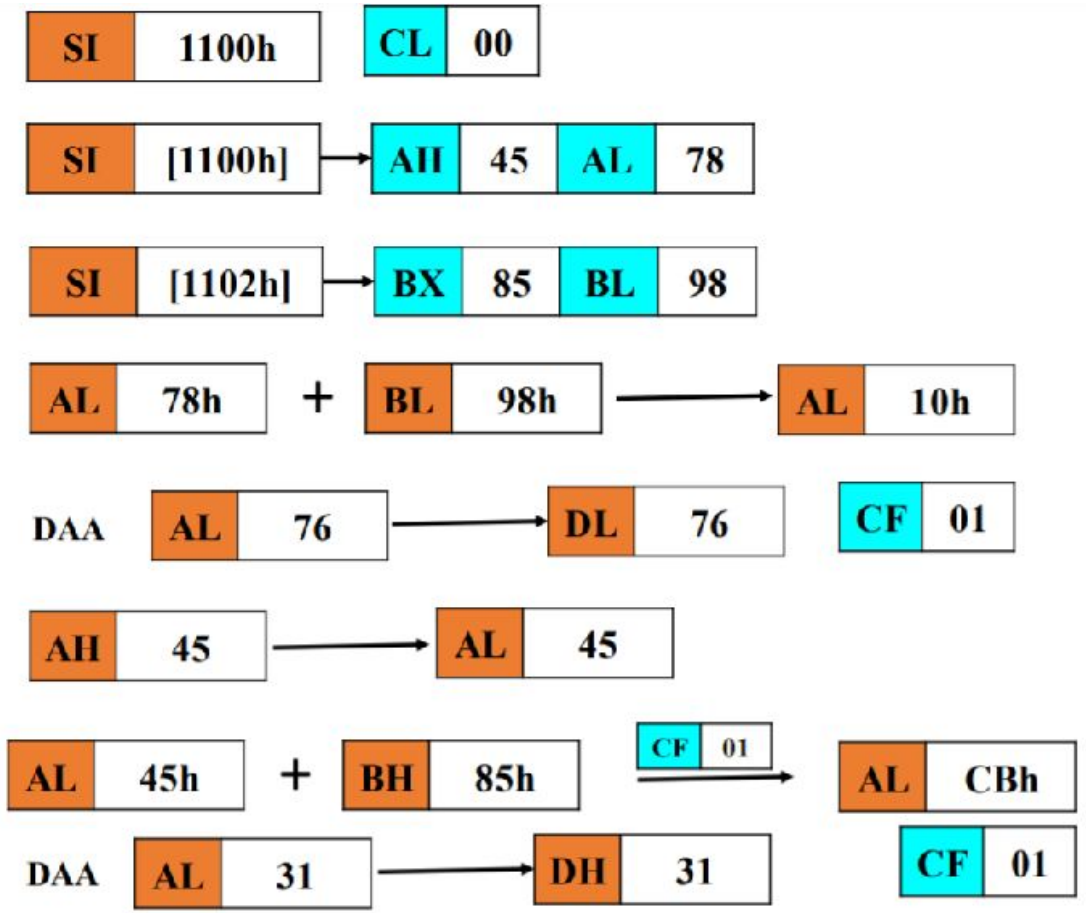
Program

```
MOV SI,1100H
MOV CL,00H
MOV AX,[SI]
MOV BX,[SI+2]
ADD AL,BL
DAA
MOV DL,AL
MOV AL,AH
ADC AL,BH
DAA
MOV DH,AL
JNC jump
INC CL
jump:
MOV [SI+4],DX
MOV [SI+6],CL
HLT
```

Example

INPUT	
Memory Address	Content
1100	78
1101	45
1102	98
1103	85

OUTPUT	
Memory Address	Content
1104	76
1105	31
1106	01



Example

INPUT	
Memory Address	Content
1100	78
1101	45
1102	98
1103	85

OUTPUT	
Memory Address	Content
1104	76
1105	31
1106	01