Assembly Language Report (Week 7)

Group 7

104502517戴逸任

104502518劉冠聲

Program CODE:

.data

ninenine BYTE 81 DUP(?)

.code

start@0 PROC

mov ecx,9 ;loop count = 9

mov esi,OFFSET ninenine ;esi point to ninenine

L:

mov bh,10 ;store 10 into bh

sub bh,cl ;bh=bh-cl

mov bl,1 ;bl start by value 1

L1:

movzx eax,bh ;store bh value(1~9) into eax

mul bl ;ax=ax\*bl

mov [esi],al ;store the value into ninenine

inc esi ;move esi to next address

inc bl ;bl=bl+1

cmp bl,10 ;compare b1 and 10

jnz L1 ;jump to L1 until b1=10

loop L ;loop L

exit ;end

Program Step & Register state:

mov ecx,9

mov esi,OFFSET ninenine

L:

mov bh,10 ;bh=10

sub bh,cl ;bh=1~9

mov bl,1 ;bl=1

L1:

movzx eax,bh ;eax=1~9

mul bl ;ax=1🡪2…🡪72🡪81

mov [esi],al

inc esi ;esi=0x00404000🡪……

inc bl ;bl=1~9

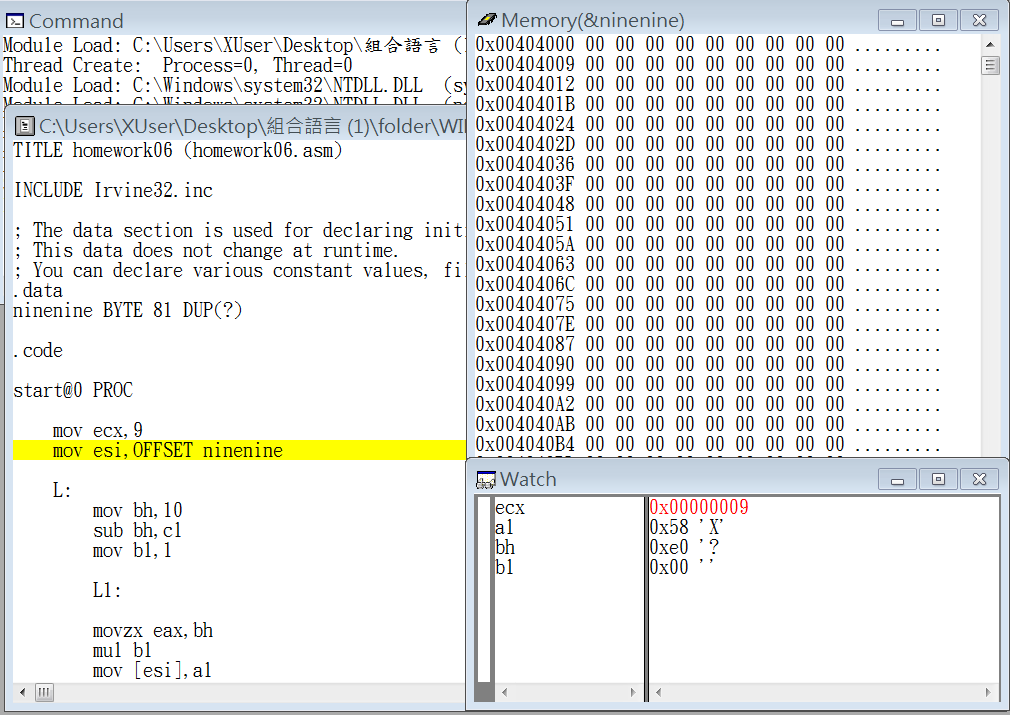
cmp bl,10 ;ZF=0 or 1

jnz L1

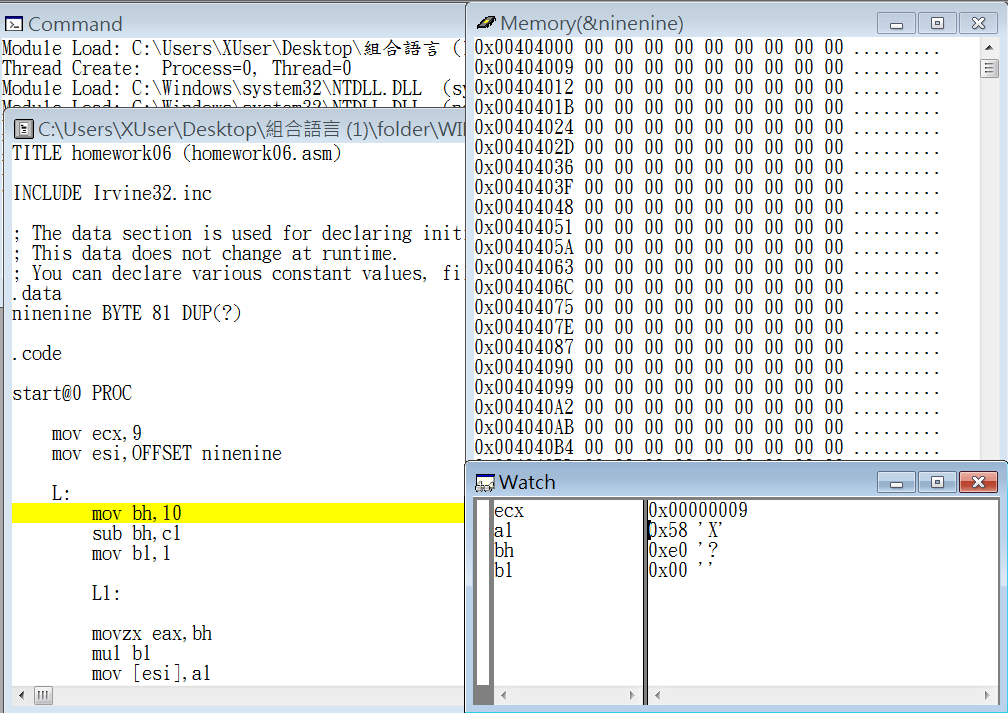
loop L ;ecx=ecx-1

exit

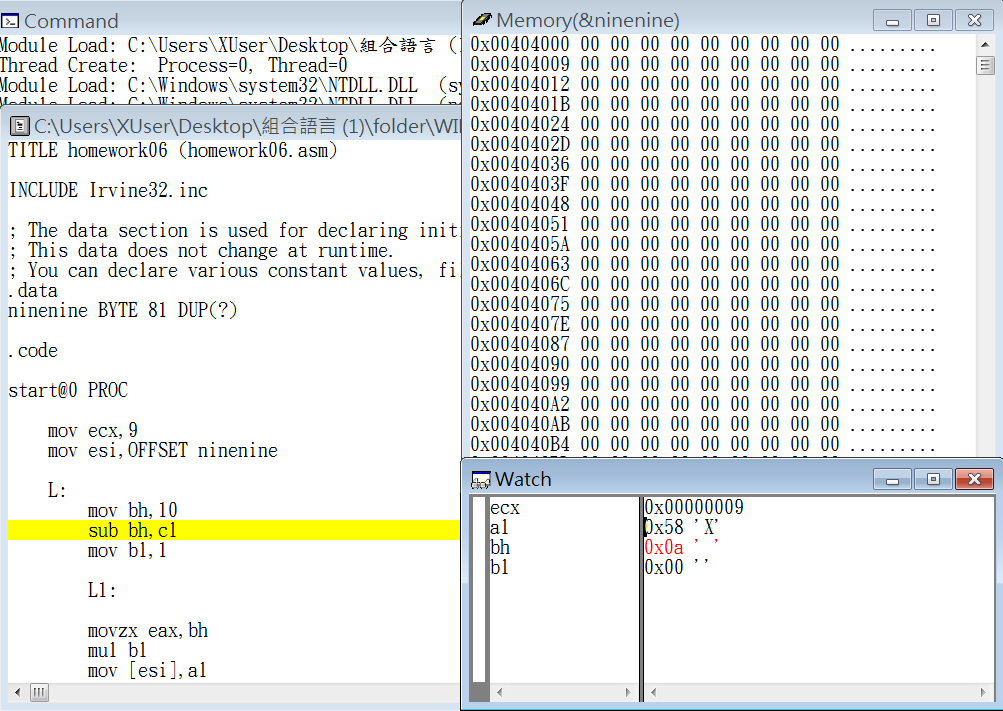
Picture & Discription:



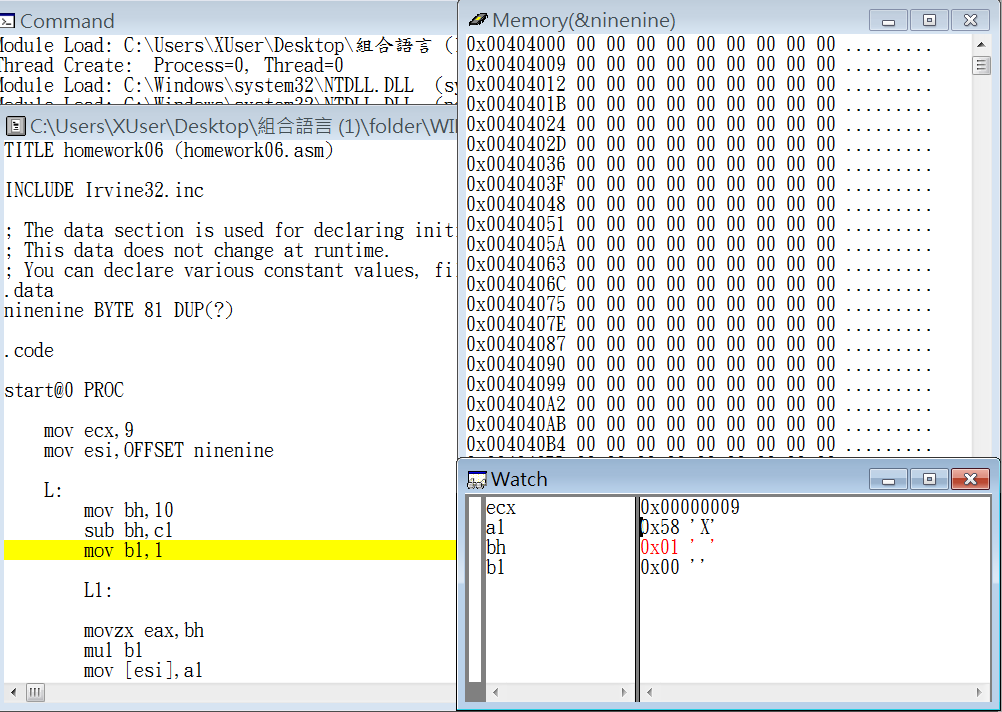
Step1: set the loop times



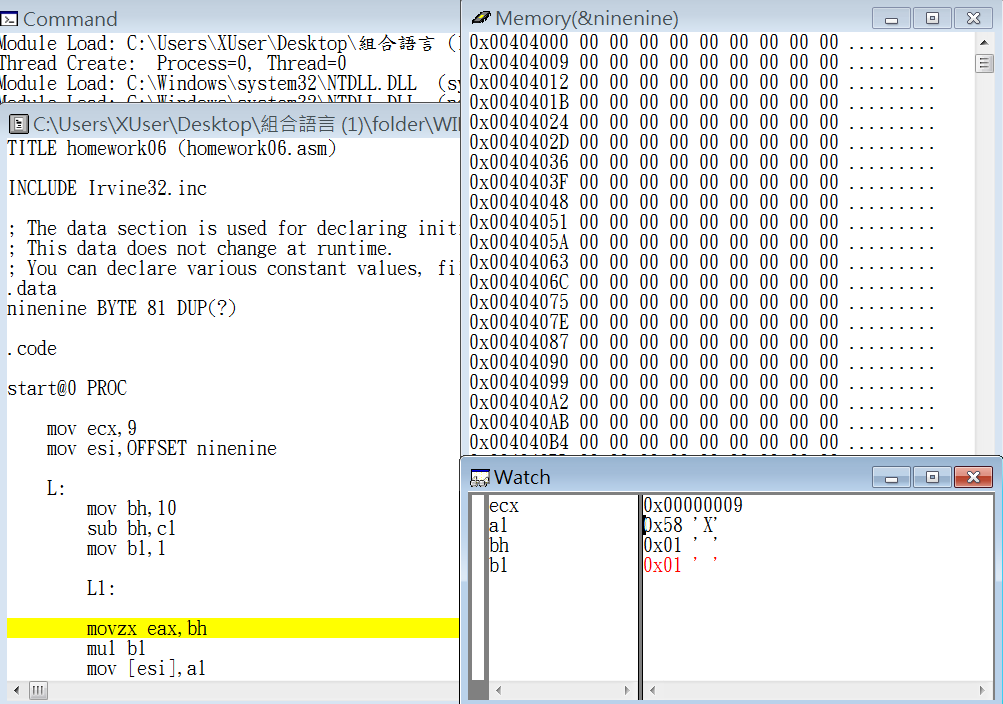
Step2: point to ninenine



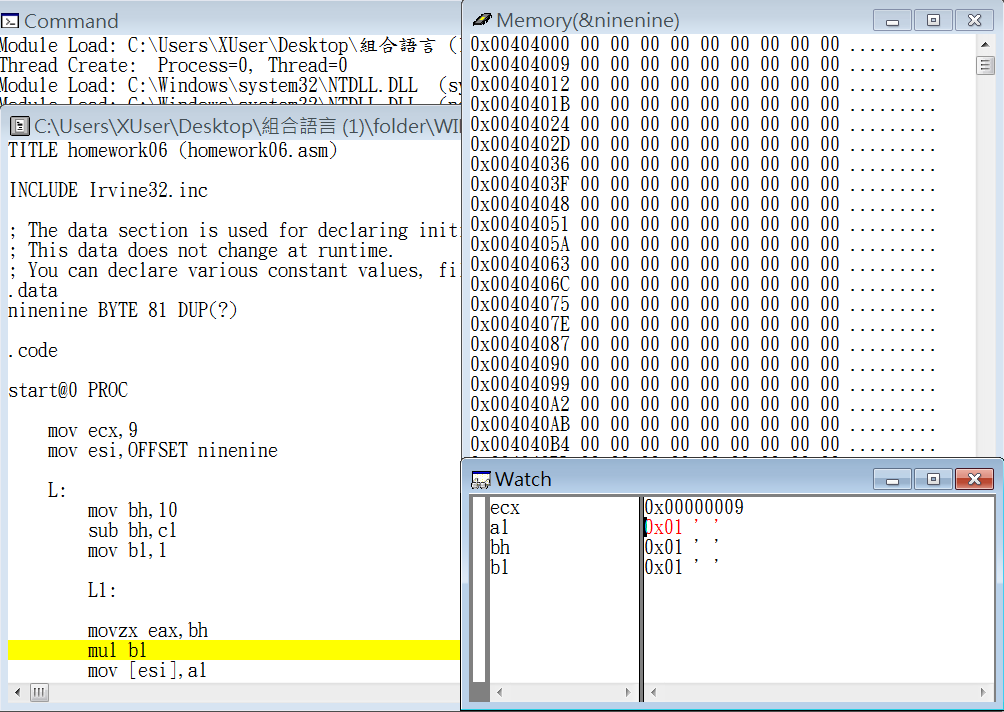
Step3: set bh = 10



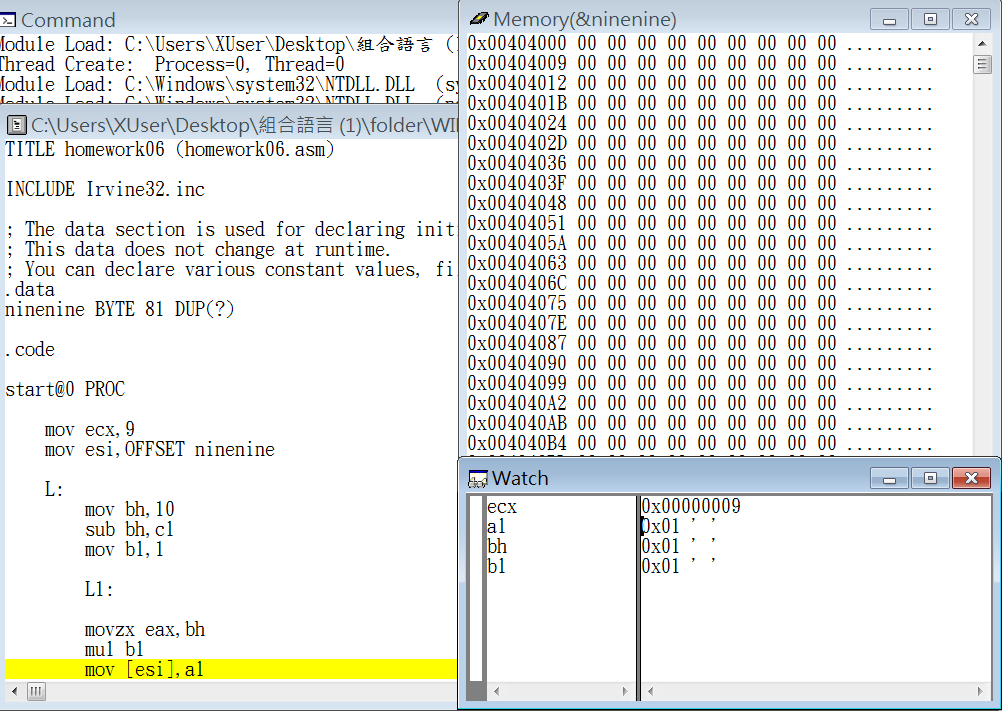
Step4: use ecx and bh to let bh be 1 to 9



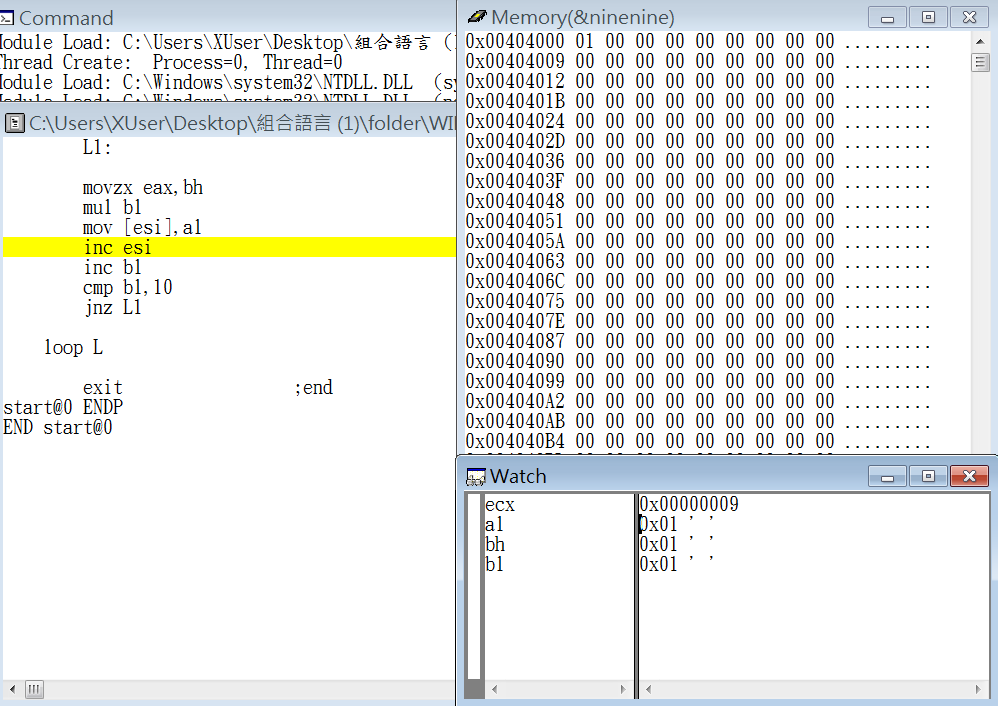
Step5: set bl = 1



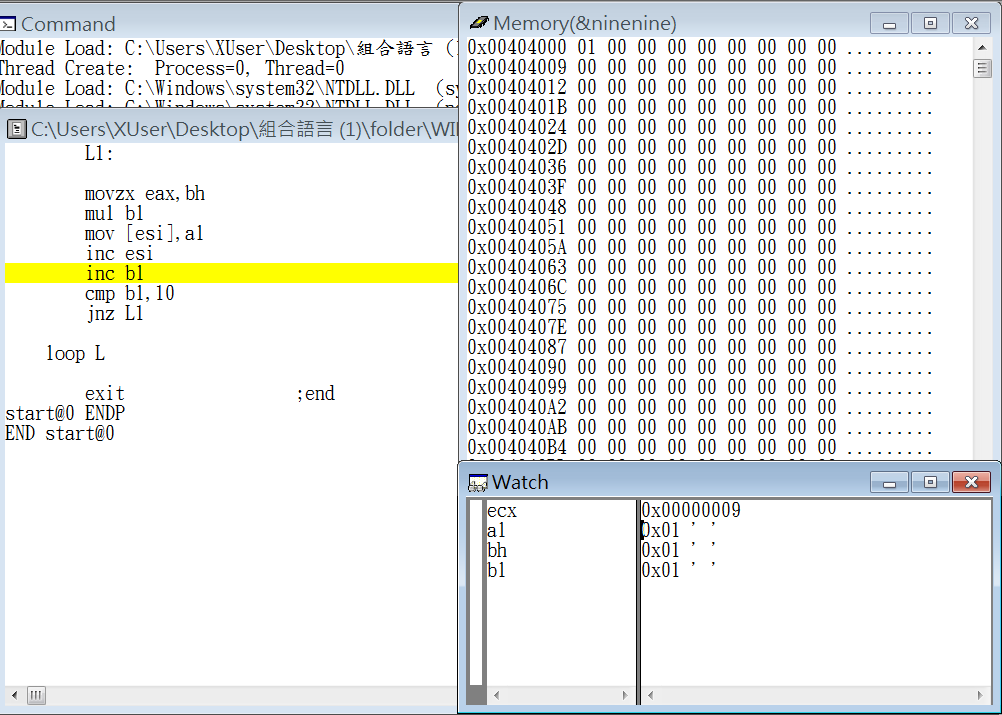
Step6: store bh value into eax to be mul



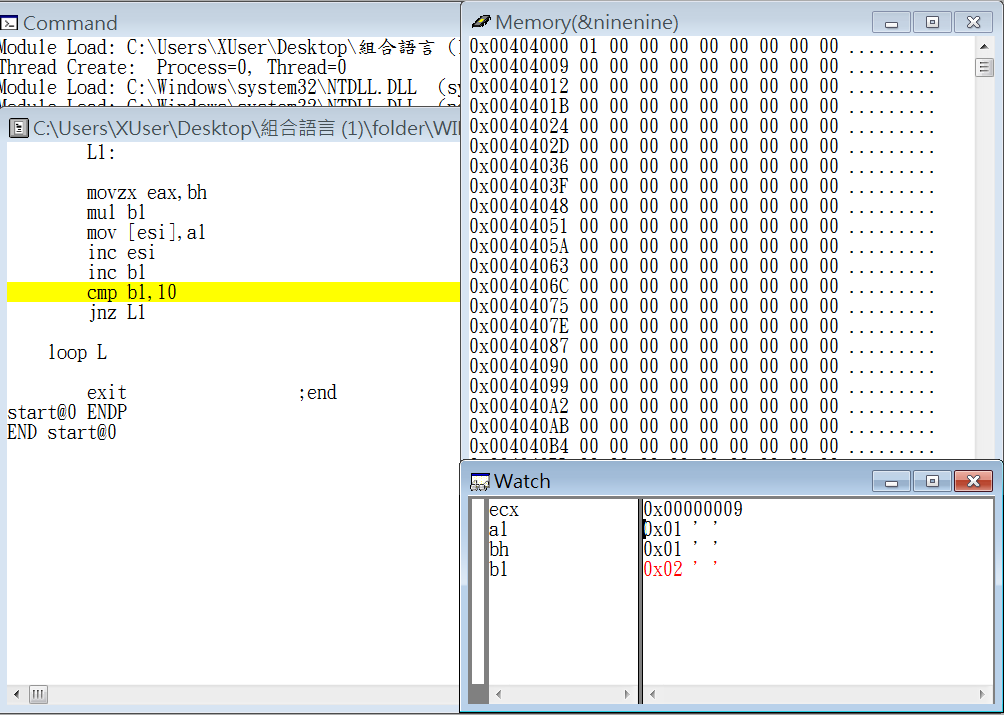
Step7: mul ax and bl and store into ax to calculate 1\*1~9\*9



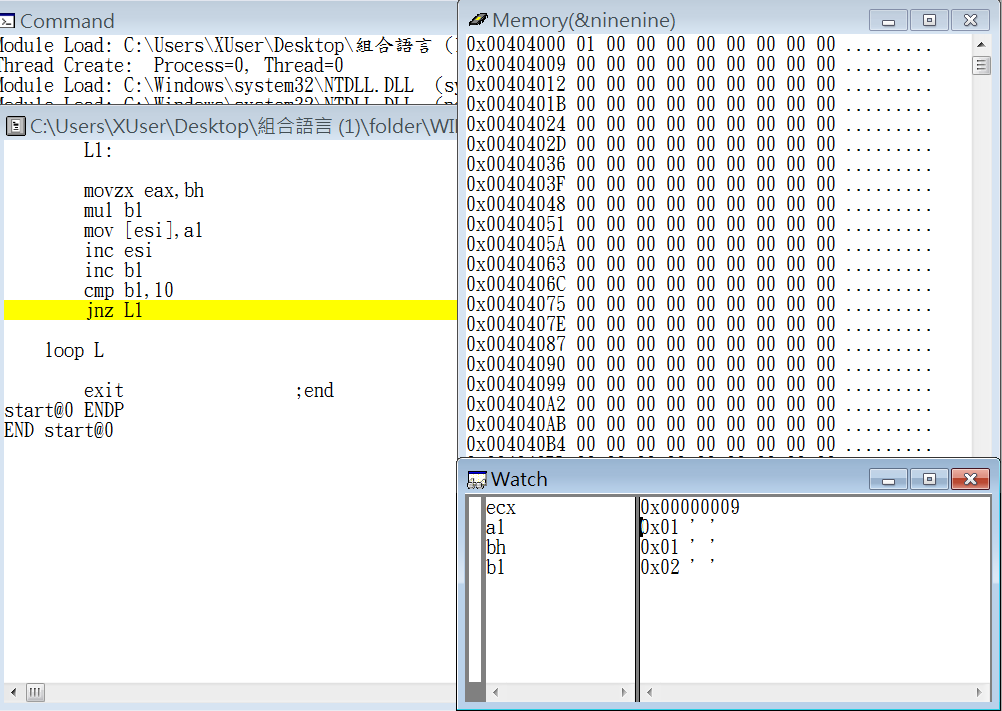
Step8: store the result into ninenine



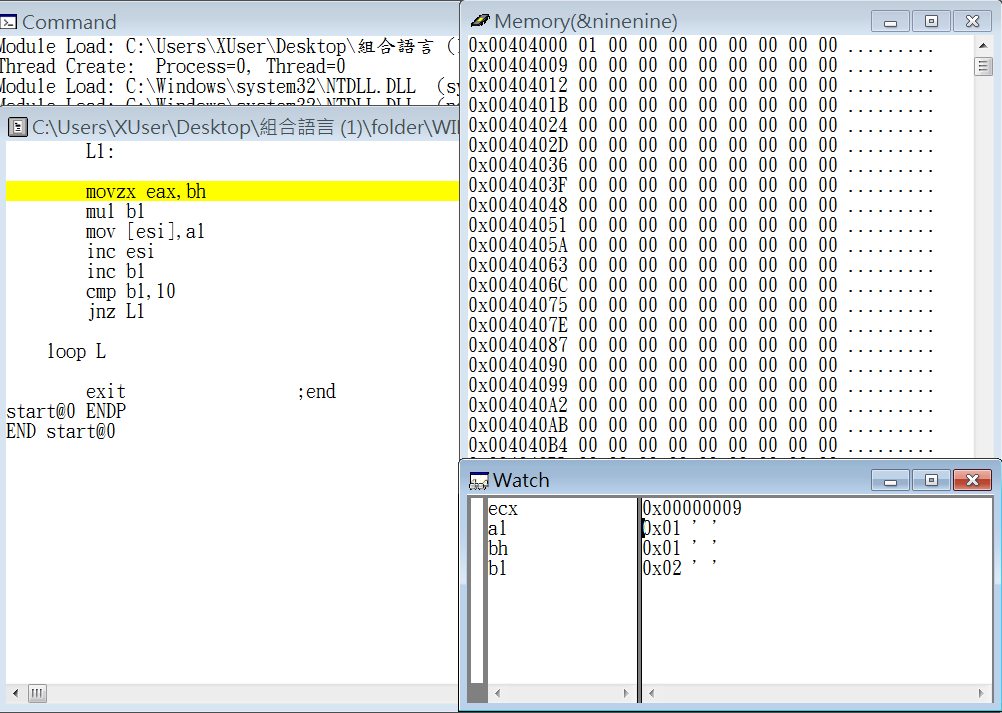
Step9: point to next memory address



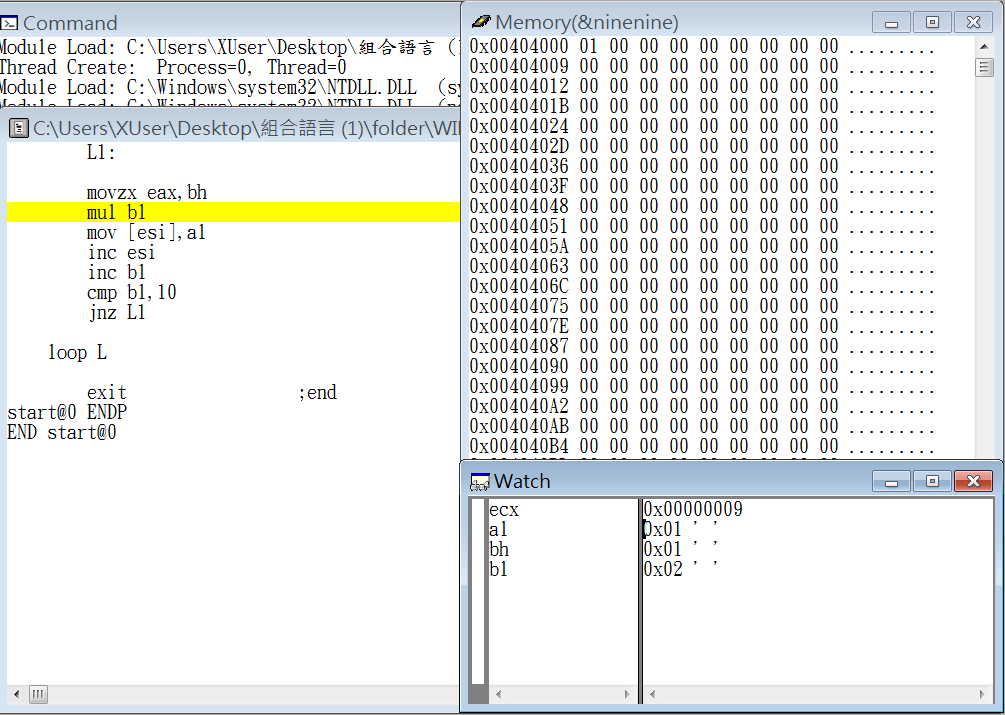
Step10: increase bl (from 1 to 10)



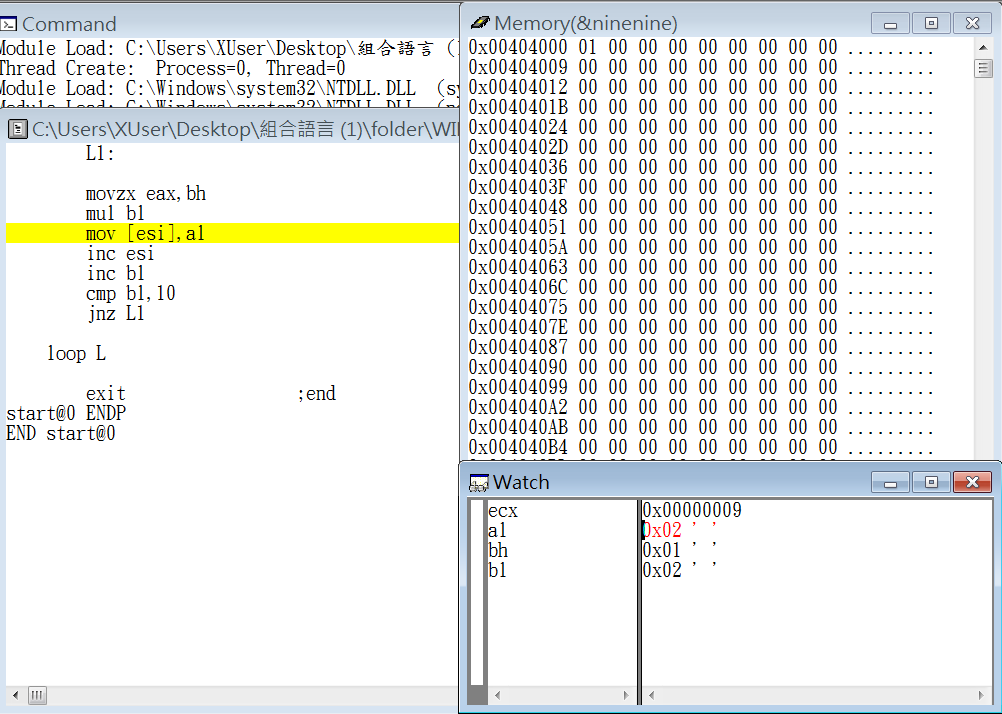
Step11: compare bl and 10,if bl = 10,zeroflag=1



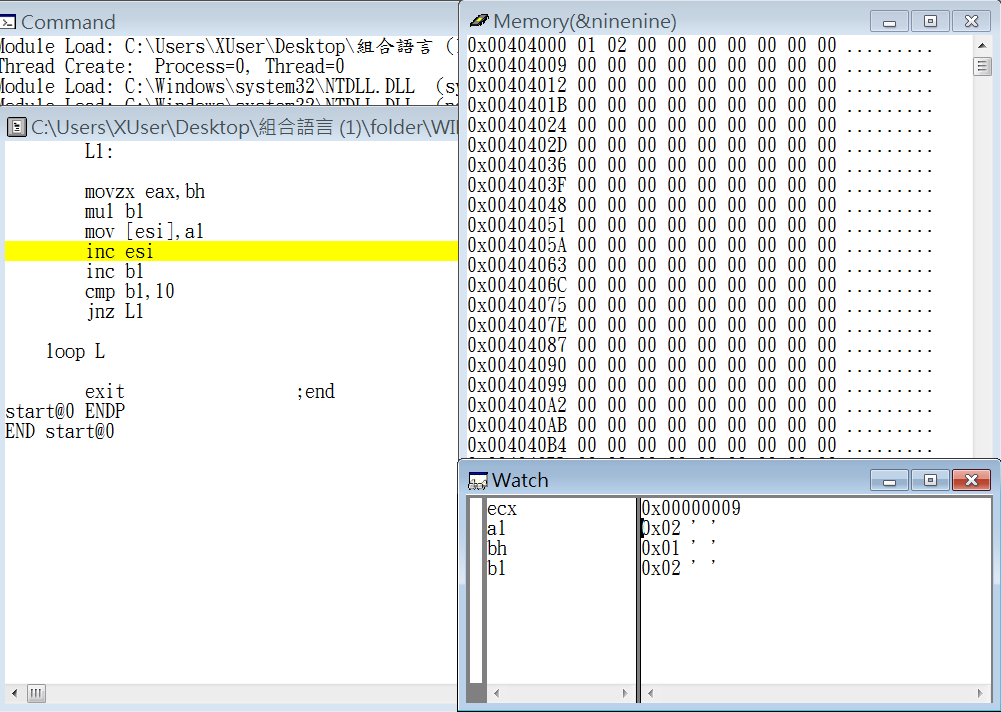
Step12: if zeroflag != 10 jump to L1 to continue multiplication



Step13: store bh value into eax to be mul



Step14: mul ax and bl and store into ax to calculate 1\*1~9\*9

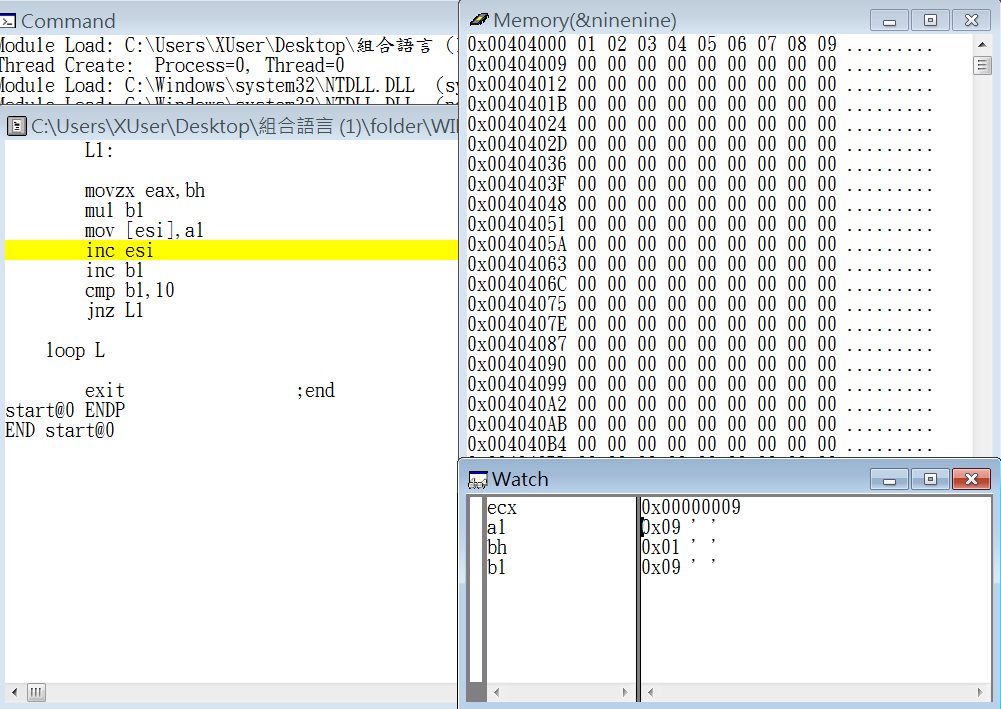


Step15: store the result into ninenine

.

.

.

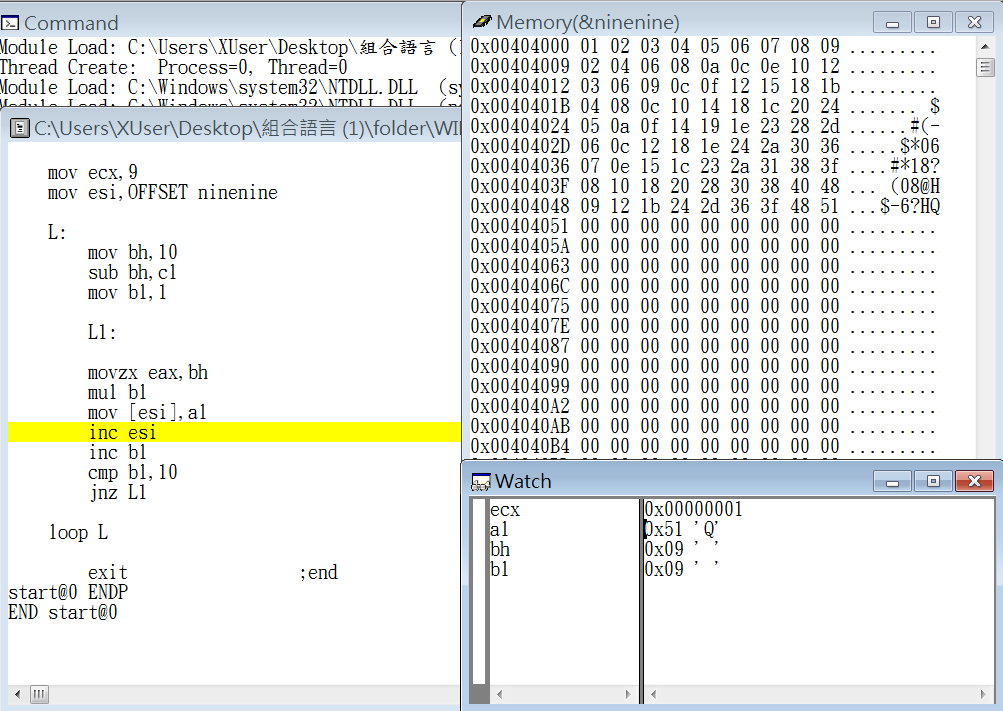


Step16: store the result into ninenine

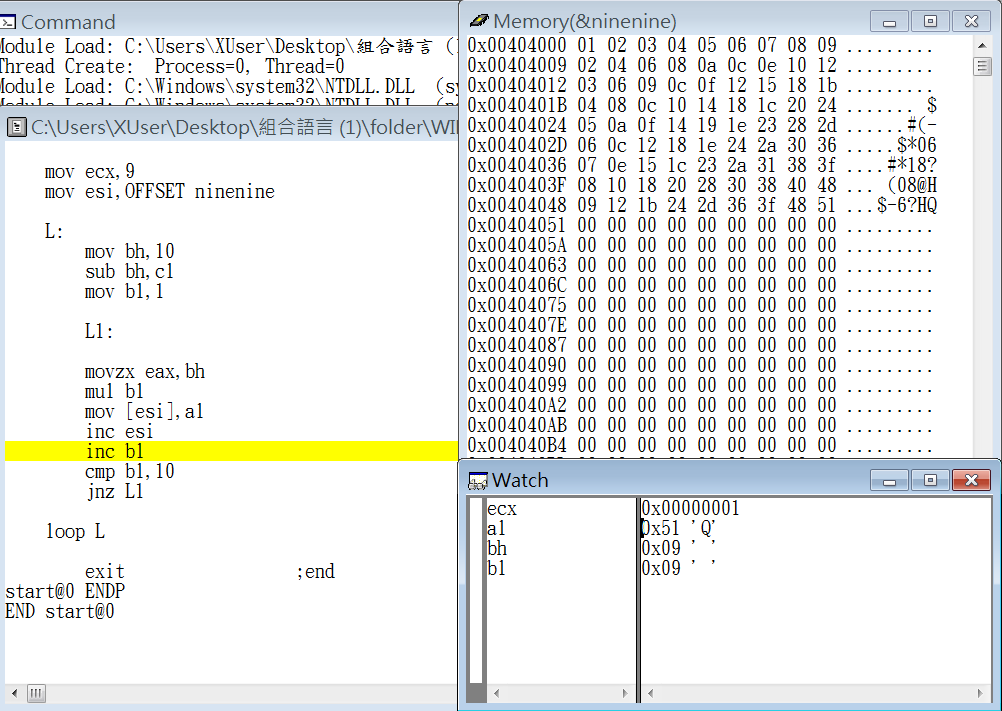
.

.

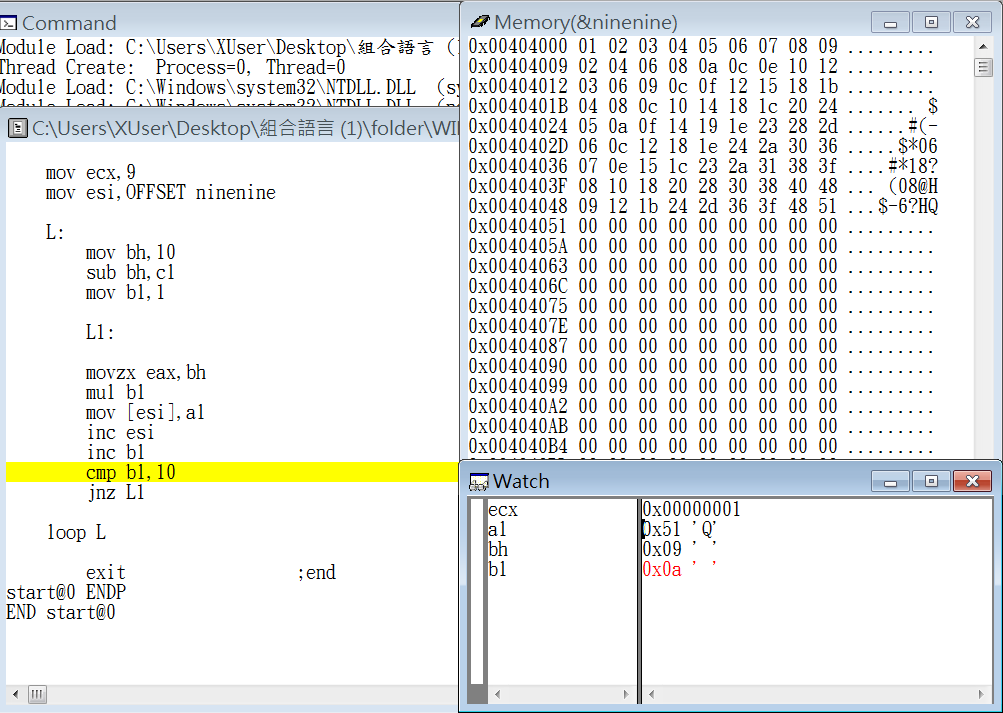
.



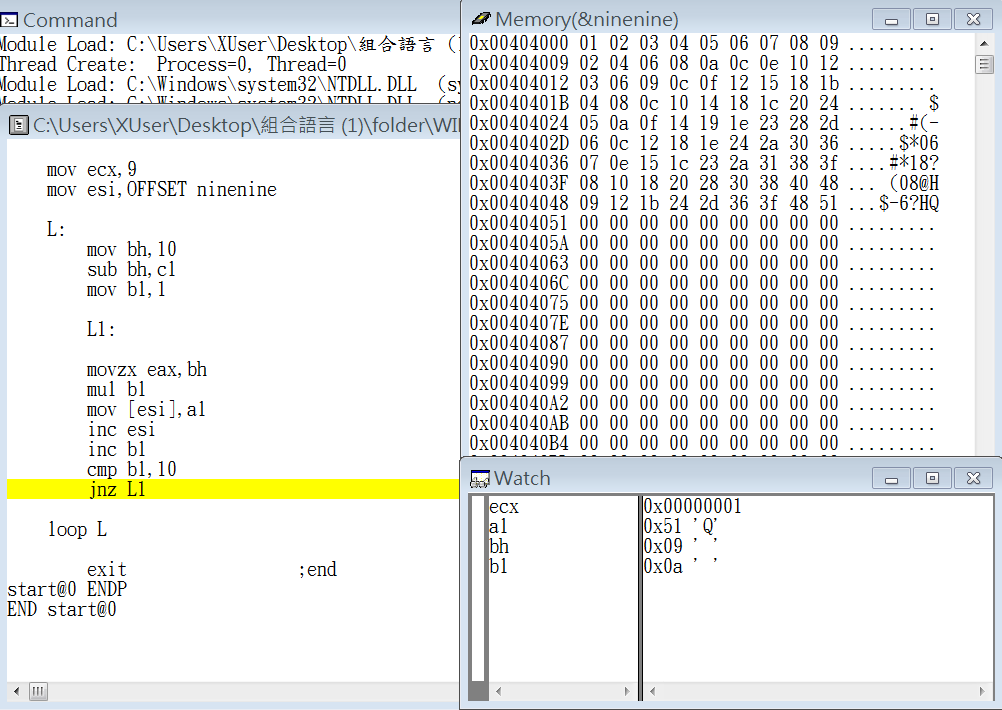
Step17: store the result into ninenine



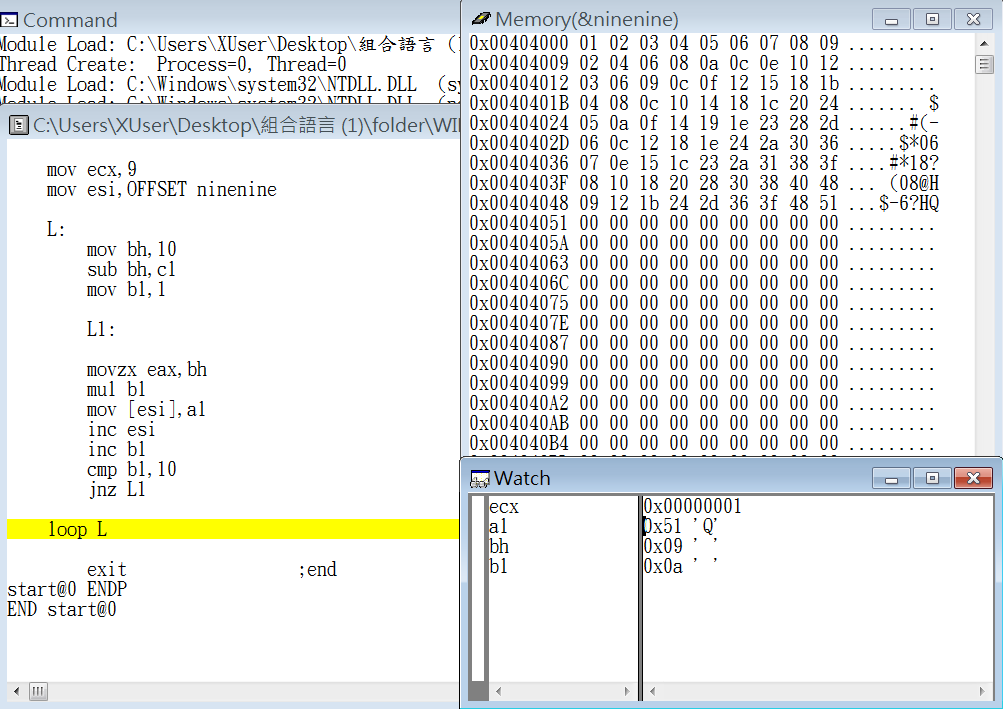
Step18: point to next memory address



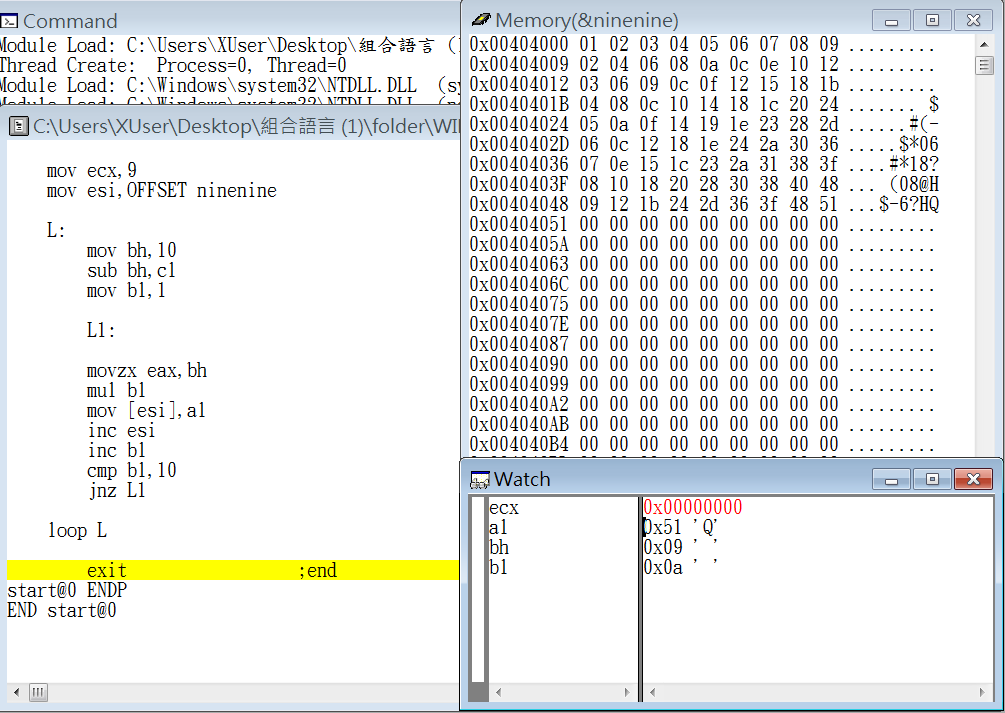
Step19: increase bl (from 1 to 10)



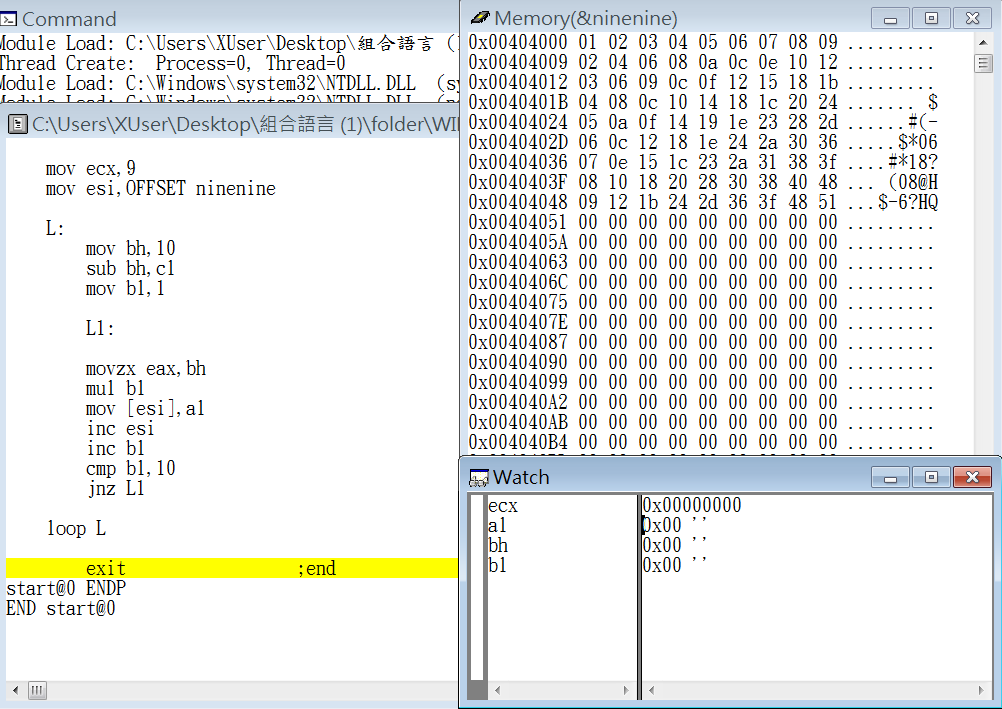
Step20: compare bl and 10,if bl = 10,zeroflag=1



Step21: if zeroflag != 10 jump to L1 to continue multiplication



Step22: ecx = 0, end loop



Step23: the end

Review:

The exercise this week is to have same consequence from the exercise which we done a few weeks ago, but we can not use serial addition this time, and then we have to use multiplication to finish it.

The problem this time is that the register had almost been used, and we can not use register eax,ecx and edx because they had already worked for some target. In the end, we use register bh and bl to store the value, and then we successfully finish the problem.

The process about one of the register had to store the value of horizontal direction and the other register had to store the value of vertical direction was really confusing us, but it was also a good experience for us to learn assembly language.