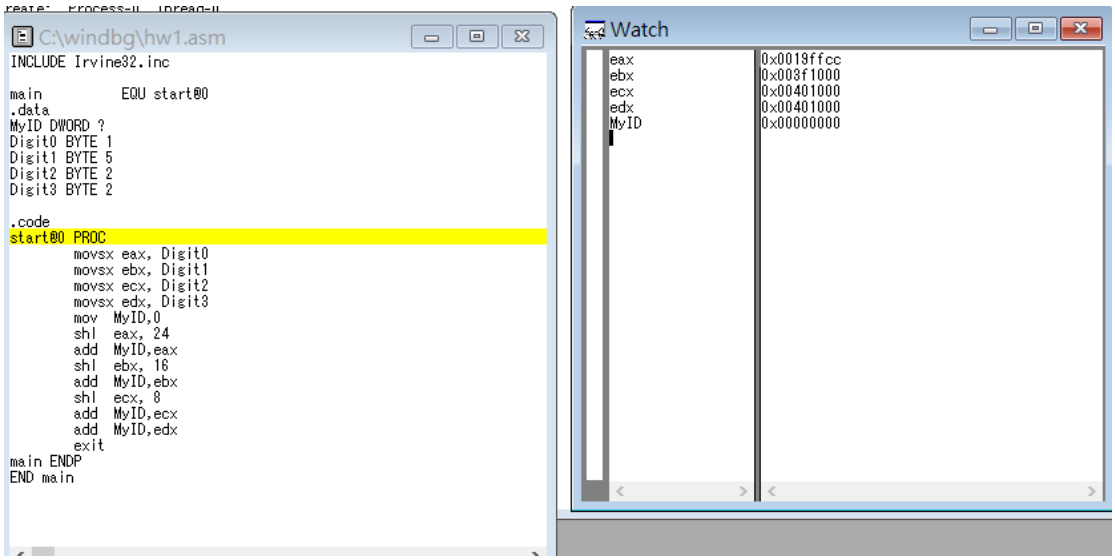
組合語言與系統程式 HW1#ARITHMETIC

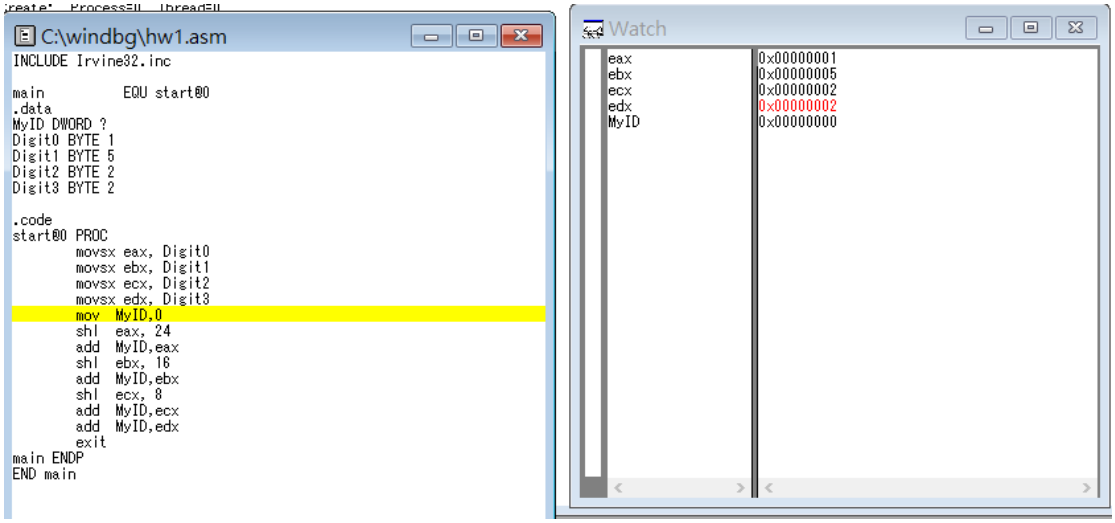
數學 4A107201522 蘇柏瑜

1 程式流程截圖、程式碼說明

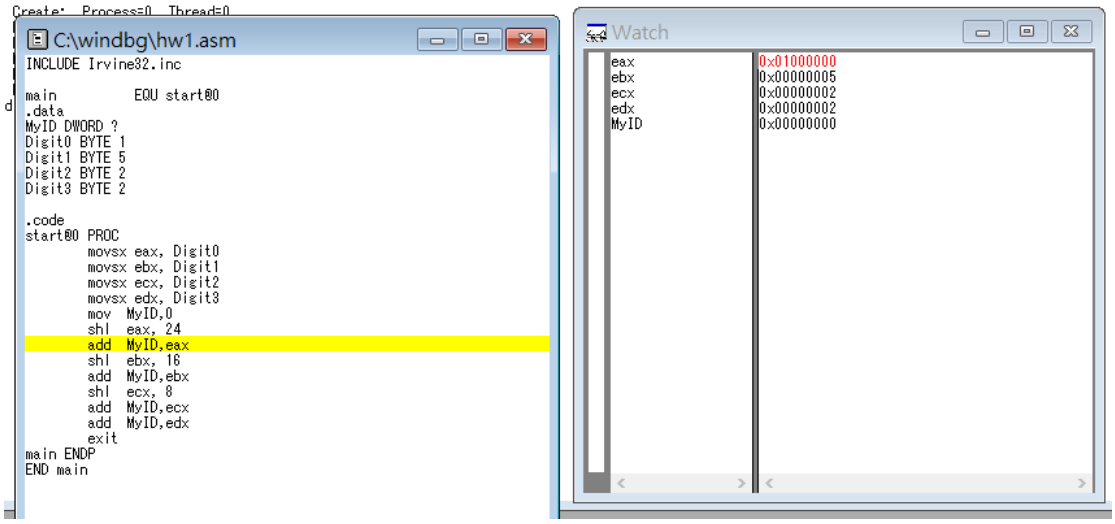
```
INCLUDE Irvine32.inc

main            EQU start@0
.data                                   宣告資料與變數
MyID DWORD ?
Digit0 BYTE 1
Digit1 BYTE 5
Digit2 BYTE 2
Digit3 BYTE 2

.code
start@0 PROC
    movsx eax, Digit0                   eax=1
    movsx ebx, Digit1                   ebx=5
    movsx ecx, Digit2                   ecx=2
    movsx edx, Digit3                   edx=2
    mov   MyID,0                        MyID=0
    shl   eax, 24                       eax=eax*24 左移動 6 個 bit
    add   MyID,eax                      MyID=MyID+eax
    shl   ebx, 16                       eax=eax*24 左移動 6 個 bit
    add   MyID,ebx                      MyID=MyID+ebx
    shl   ecx, 8                        eax=eax*24 左移動 4 個 bit
    add   MyID,ecx                      MyID=MyID+ecx
    add   MyID,edx                      MyID=MyID+edx
    exit
main ENDP
END main
```



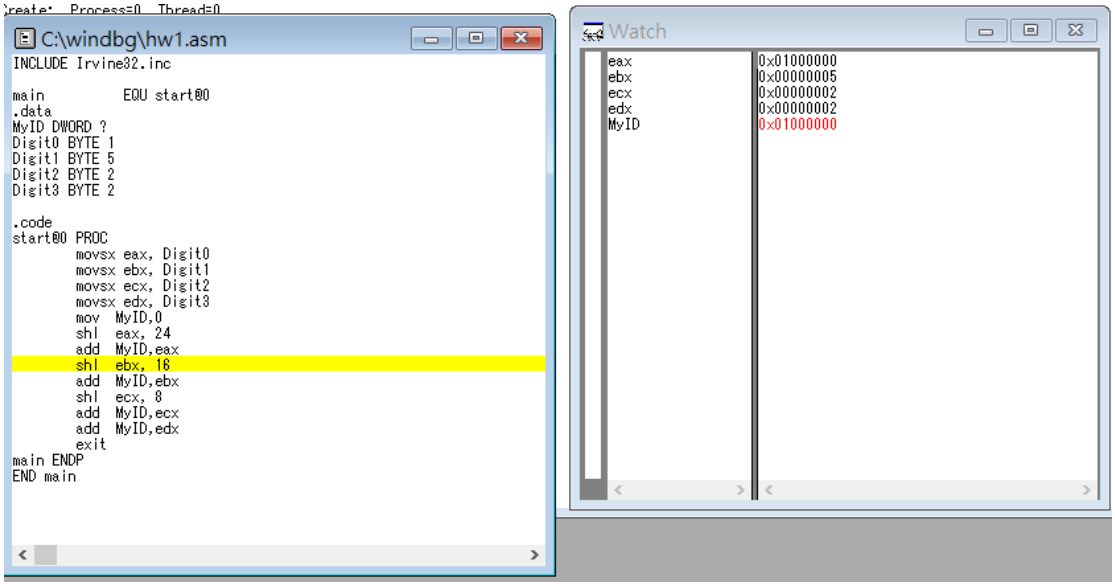
**首先把 data 及暫存器還有大小還有讀取的數值都寫好



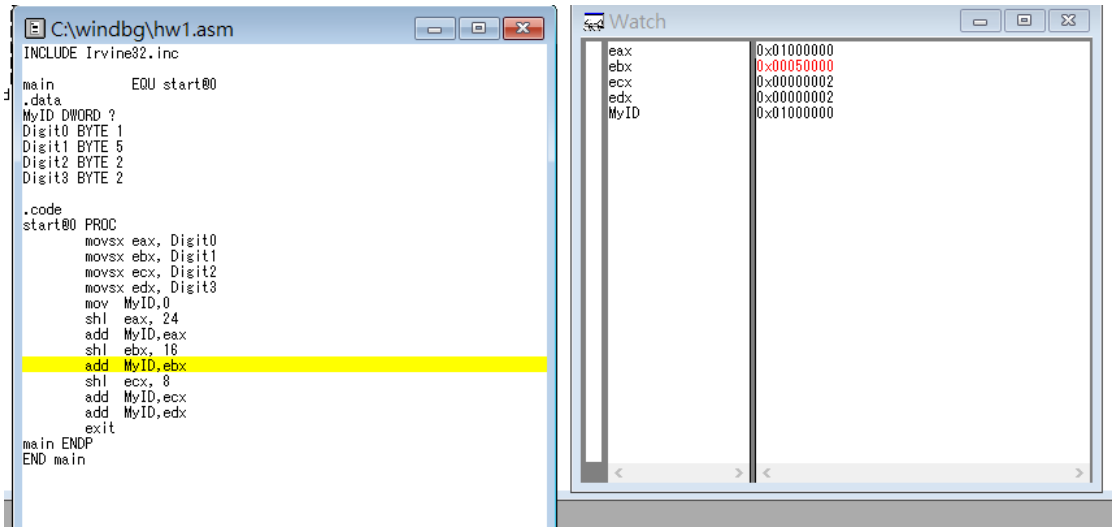
**再來就是一一把原先學生證後四碼各自讀取進去暫存器，並且確認 byte 數字滿足學生證號碼位數，並且把最終值先初始化為 0，以便之後存入



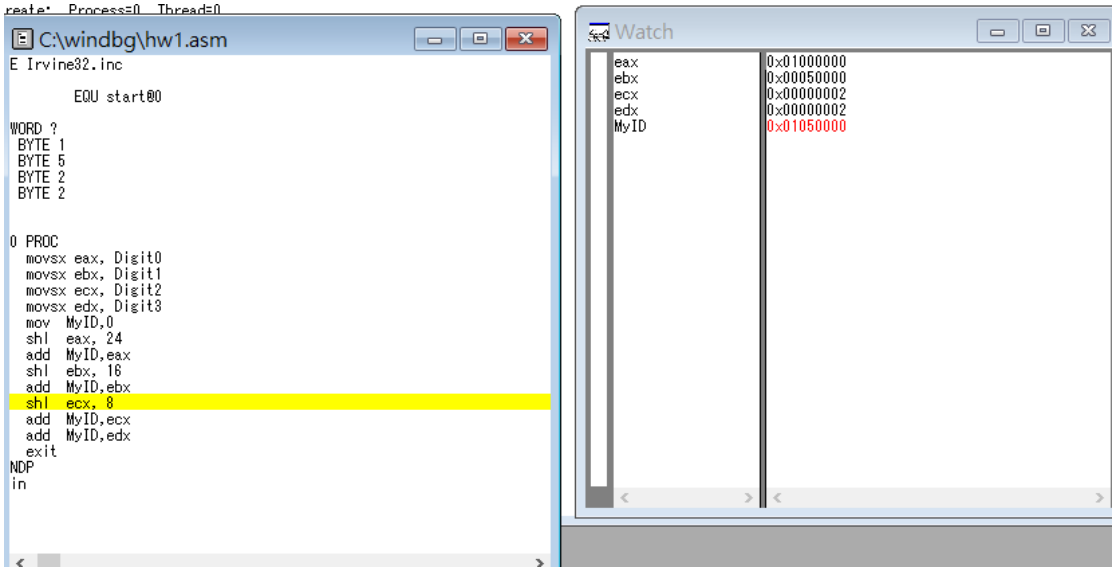
**之後把各位數字搬移到正確位址以便後面數字存入(空間左移給予下一位存入)，使用 shl 語法，並且由於 16 進位是 4bit，所以左移動 1byte*4 才是正確數值

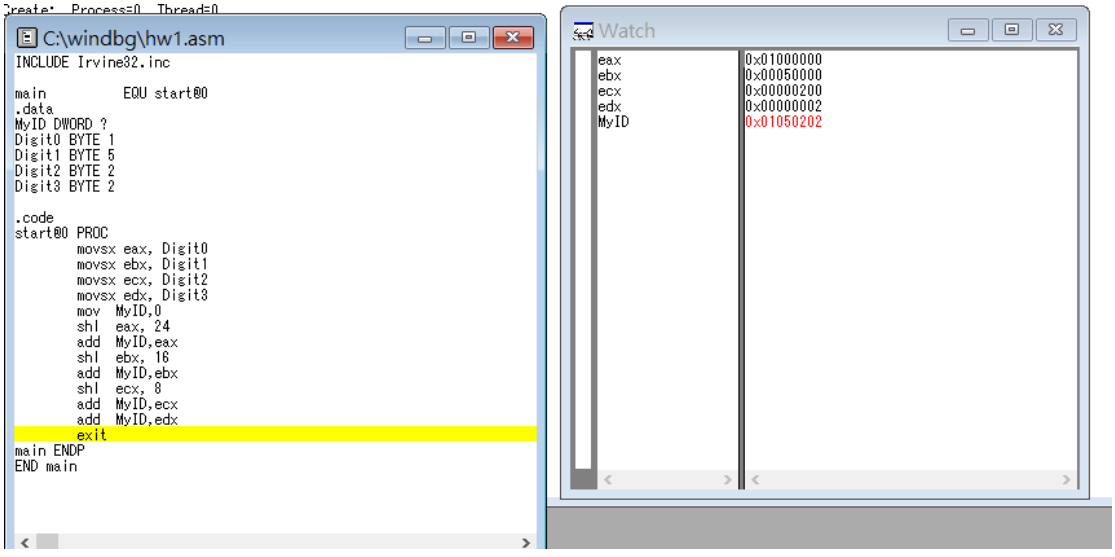


**把移動完的第一位數字搬移到暫存器 myid 好讓下一步驟進行，並同上步驟繼續操作左移動位數(shl)



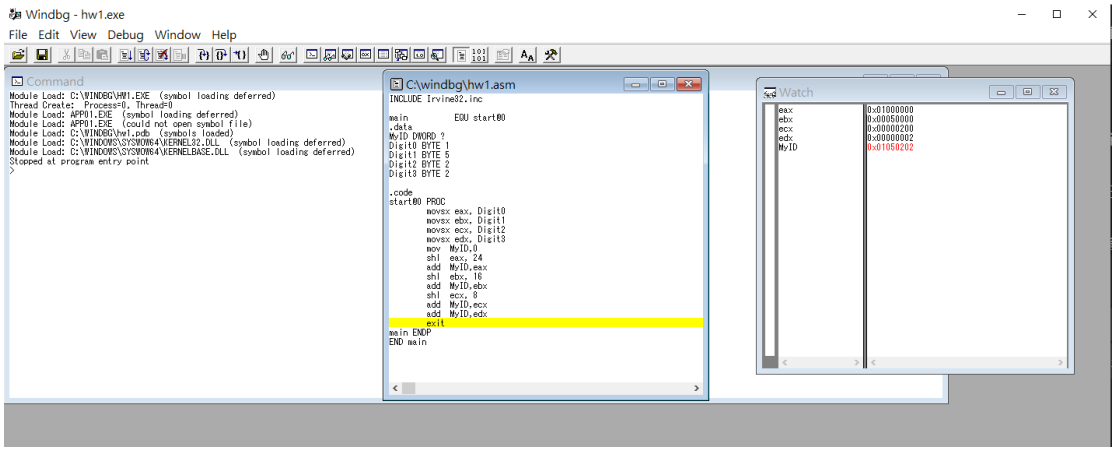
**繼續以上操作讓 eax，ebx，ecx 都能夠搬移至正確位址，並且在完成之後加入暫存器 myid





**重複把位數都存入 myid 暫存器之後就能夠得到答案與簡報檔中範例匹配

2 完成的程式畫面截圖



3 作業心得

這次可以發現以前教學提到的 little endian 排序法，這邊要使用類似 big endian 直觀輸入法，差別在於位數的移動及 byte bit 還有 16 進位的 2 進位 4 位數轉換除了空格數要注意還有一開始資料宣告還有存取暫存器的 byte 設定都顯得渺小而又重要，因為程式的小細節往往都能在主程式及最後結果有顯這影響不容忽視。