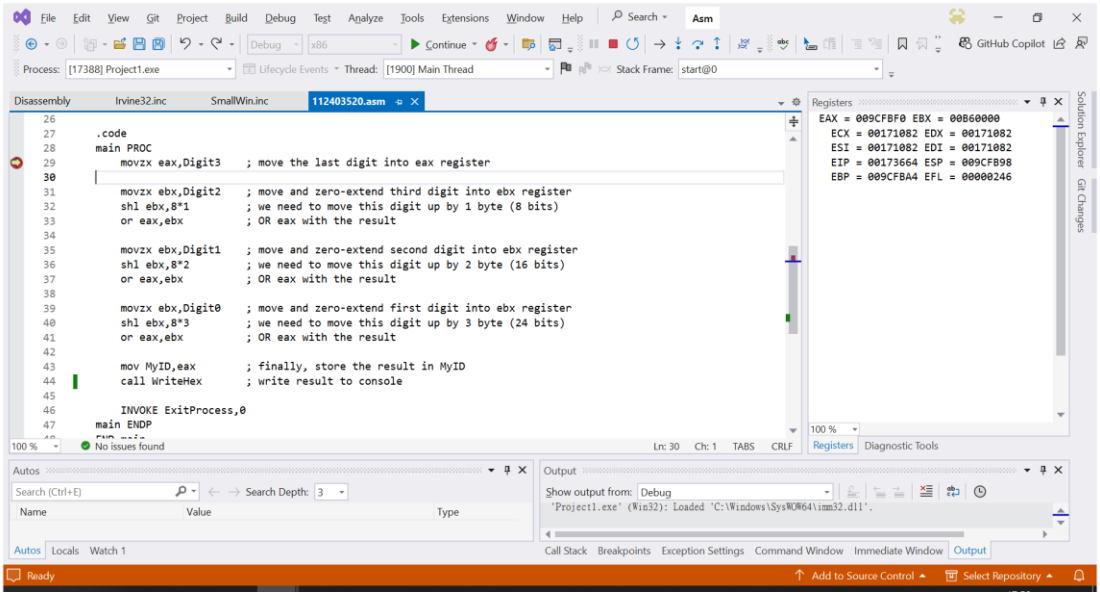
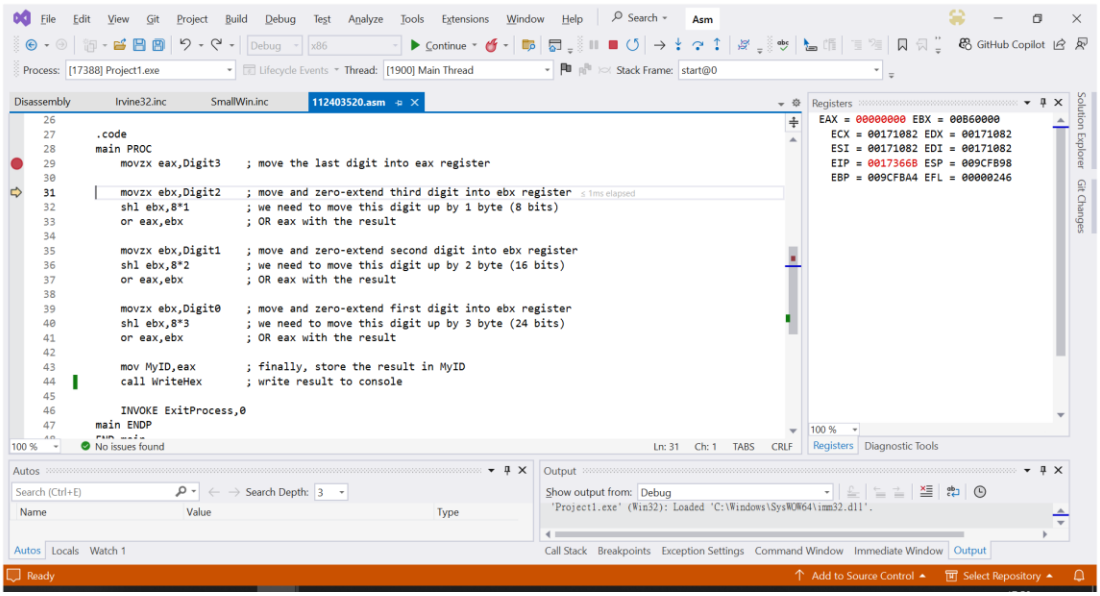
組合語言 作業一

112403520 李宥寬

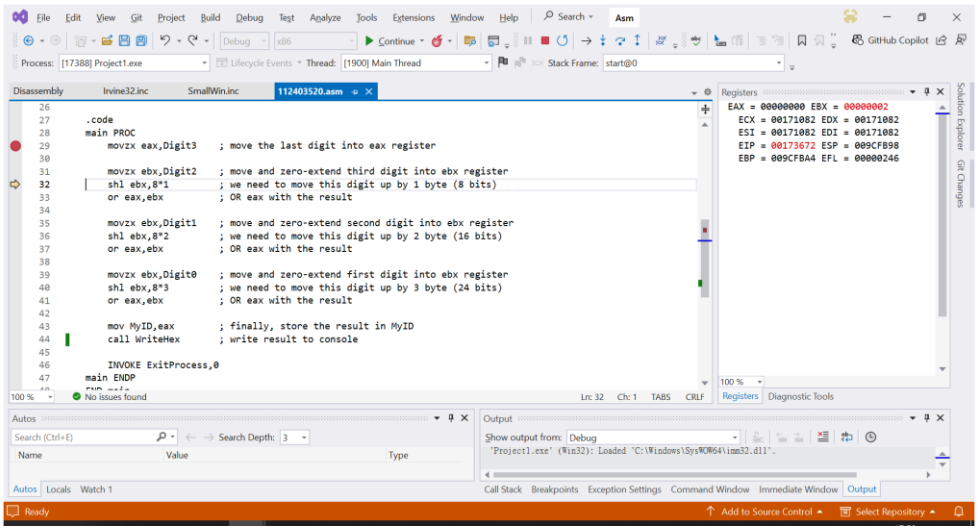
一. 程式流程截圖、程式碼說明

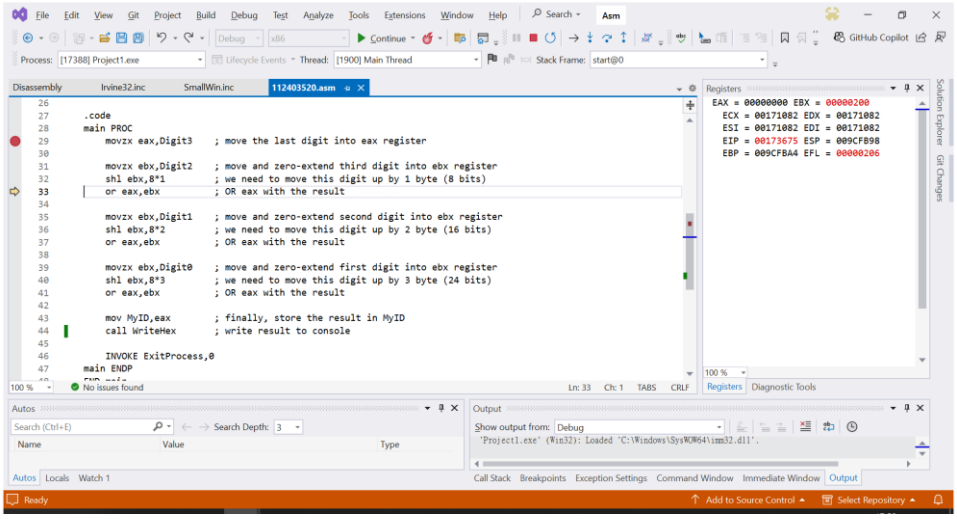


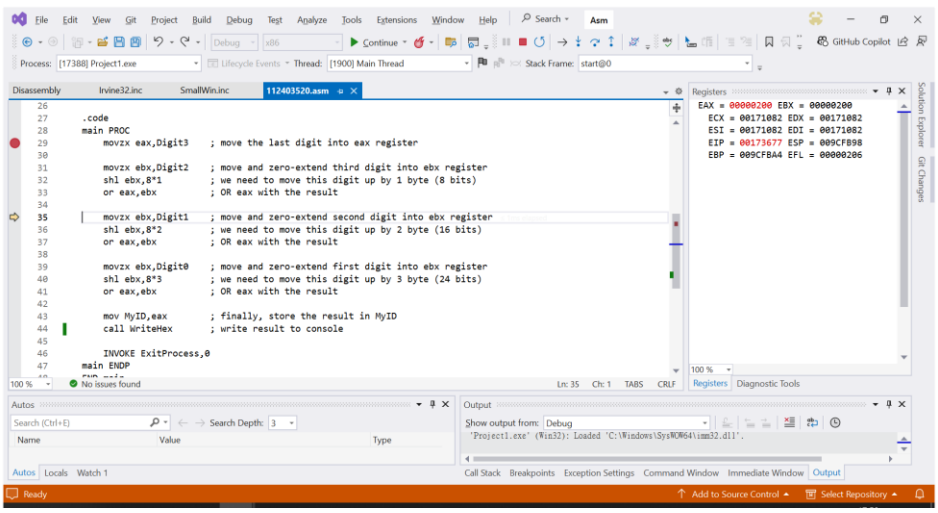
程式初始化



將第四個數字移到 eax



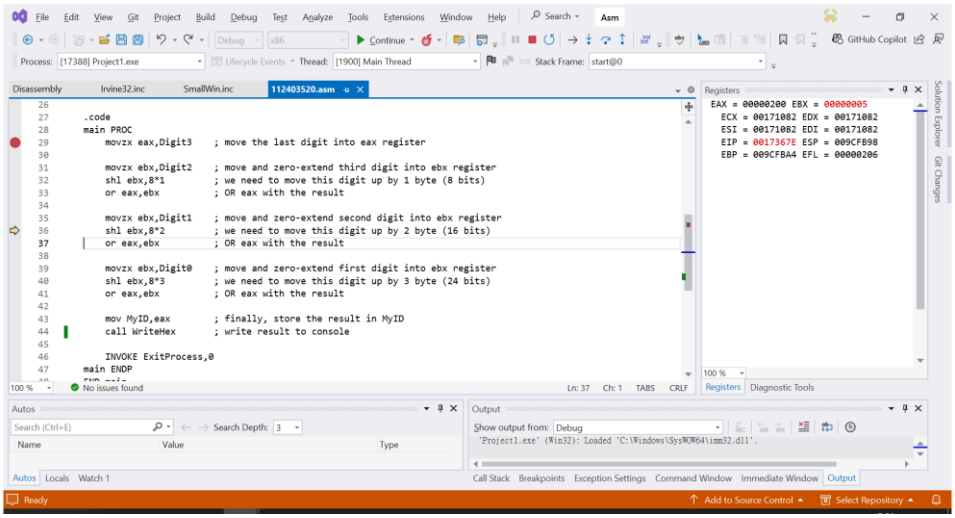


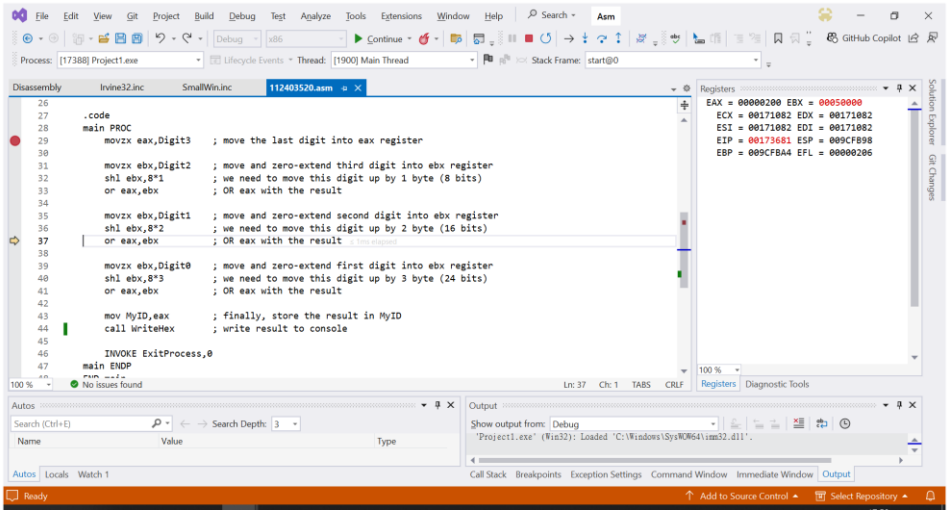


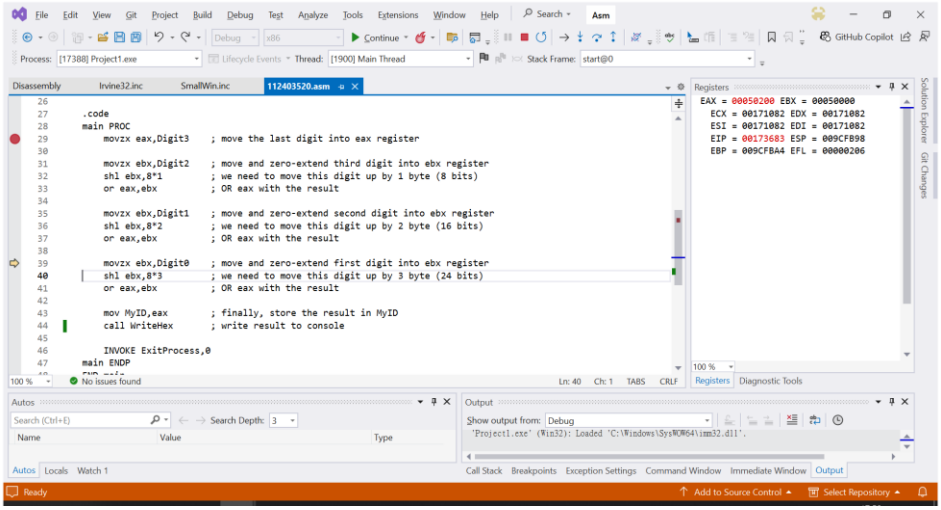
說明:

將第三個數字移到 EBX，左移 8 bit，再 OR EAX 上原本的第四位數字。

所以 EAX 由低到高的 3、4 位數就會是第三個數字，而 1、2 位還是第四個數字



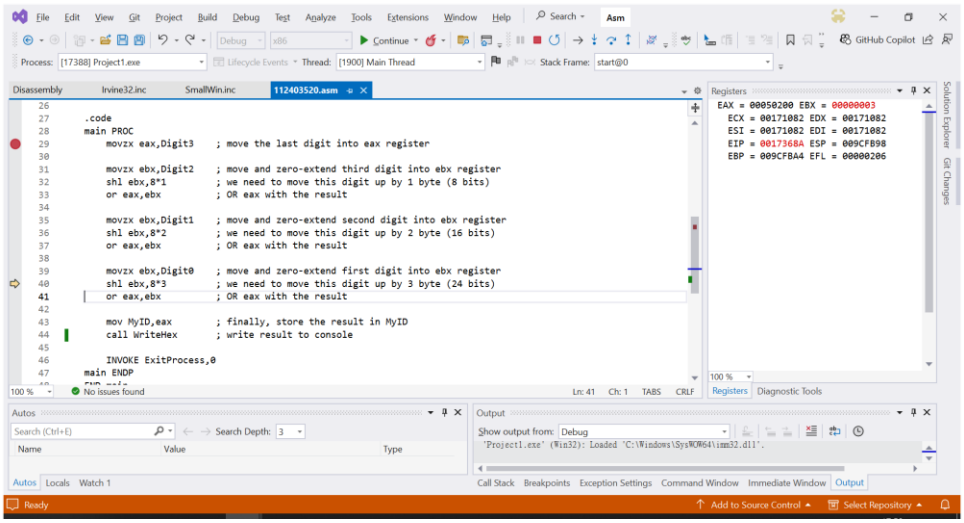


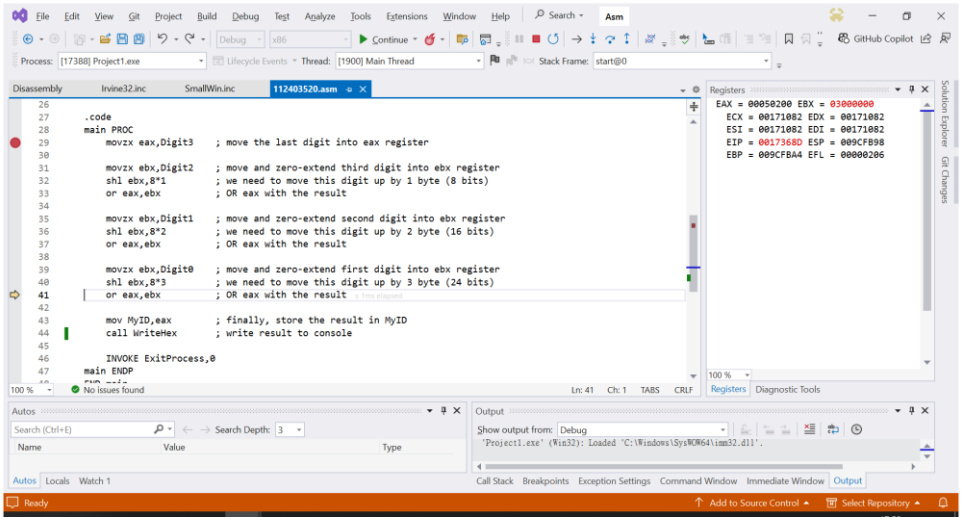


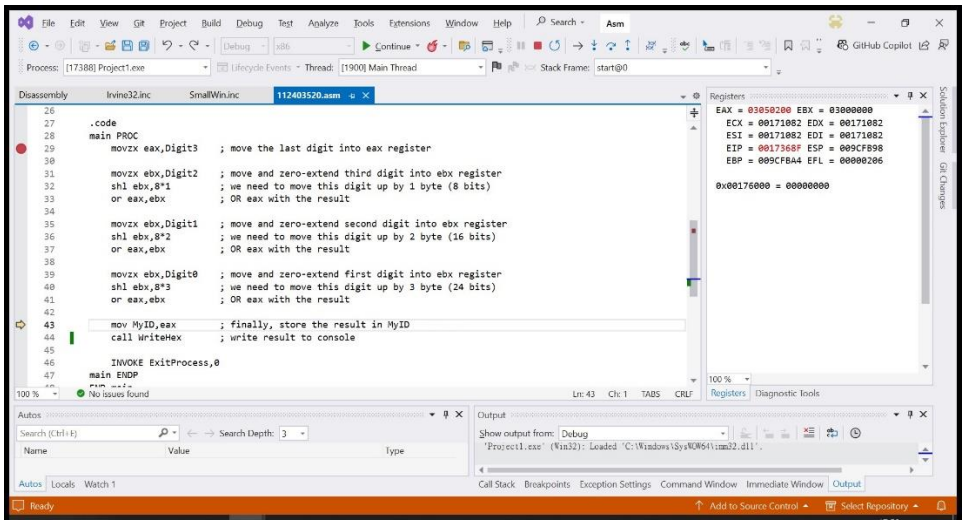
說明:

將第二個數字移到 EBX，左移 16 bit，再 OR EAX。

所以 EAX 由低到高的 5、6 位數就會是第二個數字，而 1~4 位保留原本的值



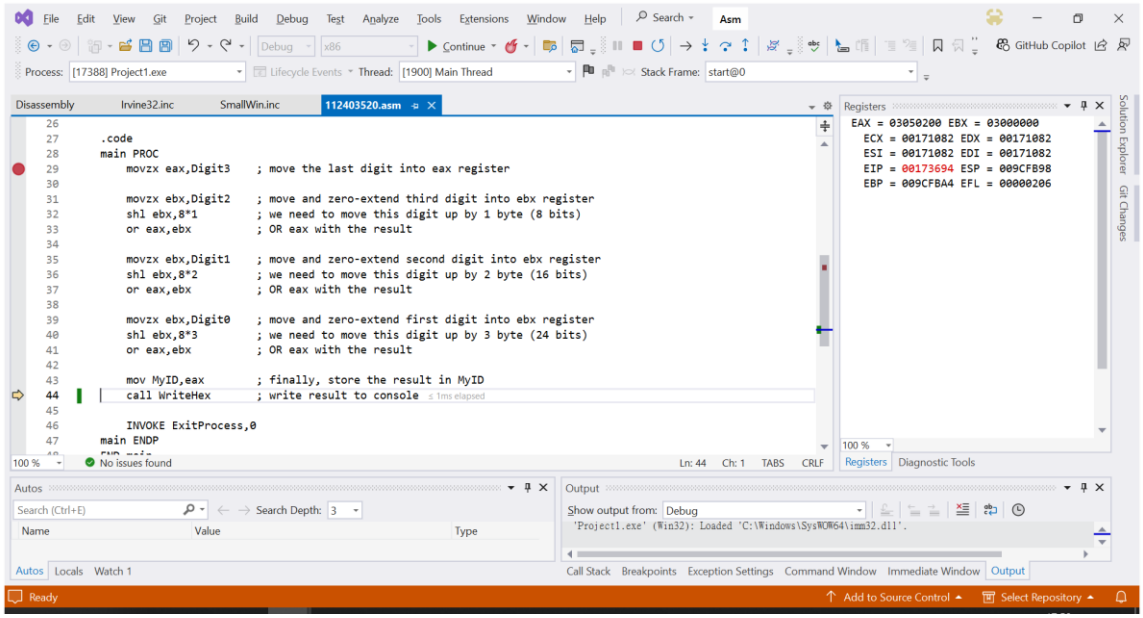


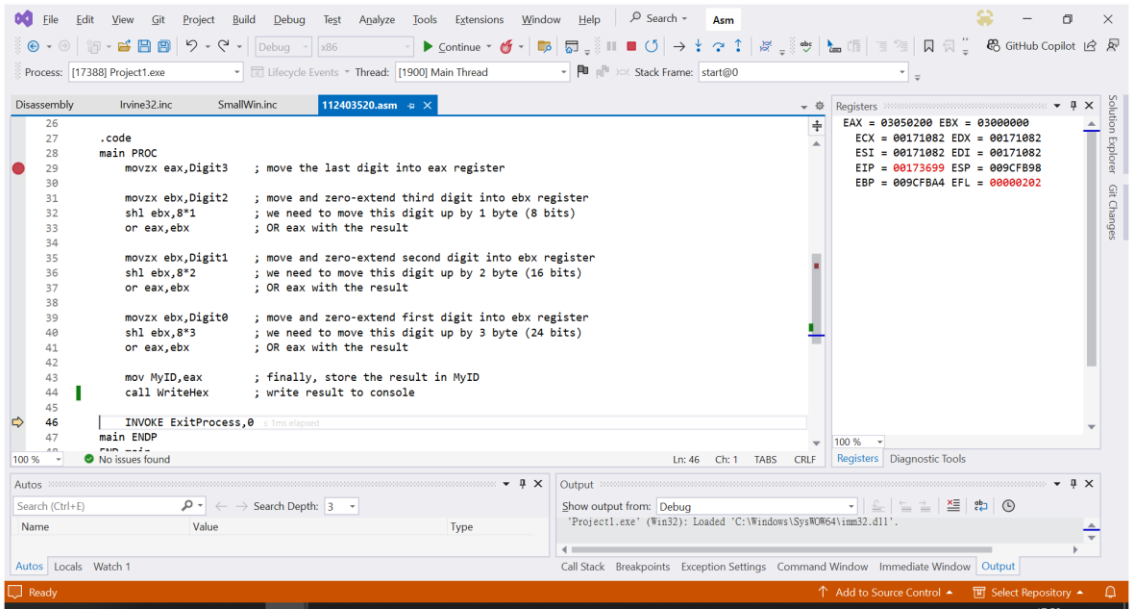


說明:

將第一個數字移到 EBX，左移 24 bit，再 OR EAX。

所以 EAX 由低到高的 7、8 位數就會是第一個數字，而 1~6 位保留原本的值

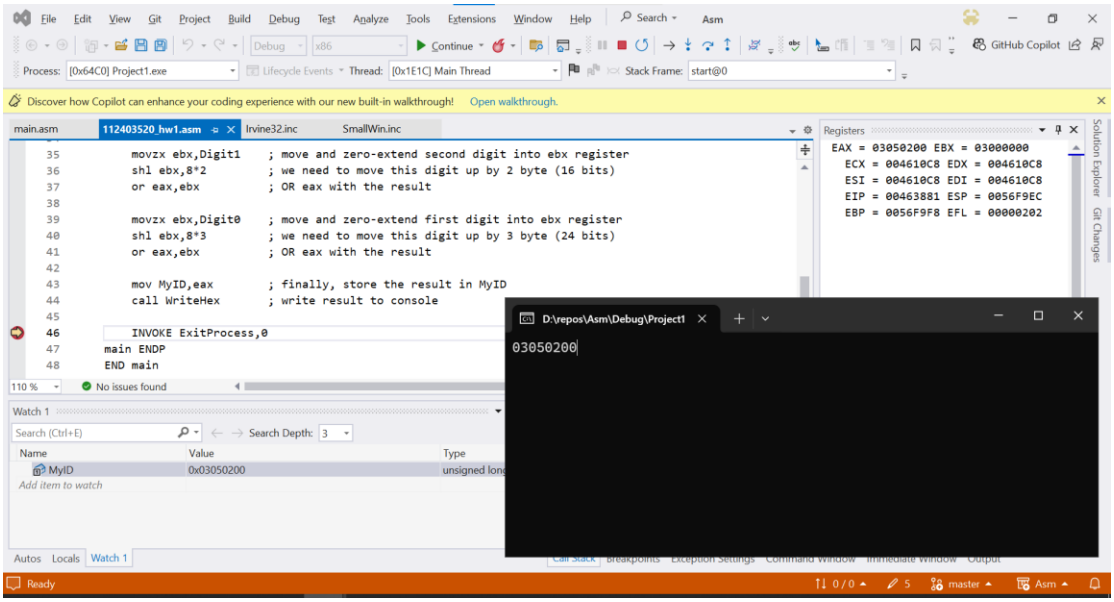




將儲存在 EAX 的結果複製到 MyID，並輸出到 Console

二. 完成的程式畫面截圖

可以看到左下角 Watch 的 MyID 和輸出到 Console 的值皆為正確的 (3520->03050200)



完整程式碼

```
;Definitions copied from SmallWin.inc:
.stack 4096
ExitProcess PROTO, dwExitCode:DWORD

.data
MyID   DWORD ?
Digit0 BYTE 3
Digit1 BYTE 5
Digit2 BYTE 2
Digit3 BYTE 0

.code
main PROC
    movzx eax,Digit3    ; move the last digit into eax register

    movzx ebx,Digit2    ; move and zero-extend third digit into ebx register
    shl ebx,8*1         ; we need to move this digit up by 1 byte (8 bits)
    or eax,ebx          ; OR eax with the result

    movzx ebx,Digit1    ; move and zero-extend second digit into ebx register
    shl ebx,8*2         ; we need to move this digit up by 2 byte (16 bits)
    or eax,ebx          ; OR eax with the result

    movzx ebx,Digit0    ; move and zero-extend first digit into ebx register
    shl ebx,8*3         ; we need to move this digit up by 3 byte (24 bits)
    or eax,ebx          ; OR eax with the result

    mov MyID,eax        ; finally, store the result in MyID
    call WriteHex       ; write result to console

    INVOKE ExitProcess,0
main ENDP
END main
```

三. 作業心得

這次的組合語言作業讓我深入學習了 bitwise OR 的運用。透過搭配 AND、SHL 和 SHR 指令，我能夠有效實現各種 bit masking 操作，精確提取所需的位元值，從而提高資料儲存和運用的效率。

此外，這次作業中我還使用了 MOVZX 指令，這個指令讓我能夠將較小資料型別的數據轉移到較大的暫存器中，同時自動在高位元補零，這樣的特性不僅提高了程式的可讀性，也大大簡化了數據處理的過程。這些學習經驗不僅增強了我對組合語言的理解，也提升了我在資料處理方面的能力。