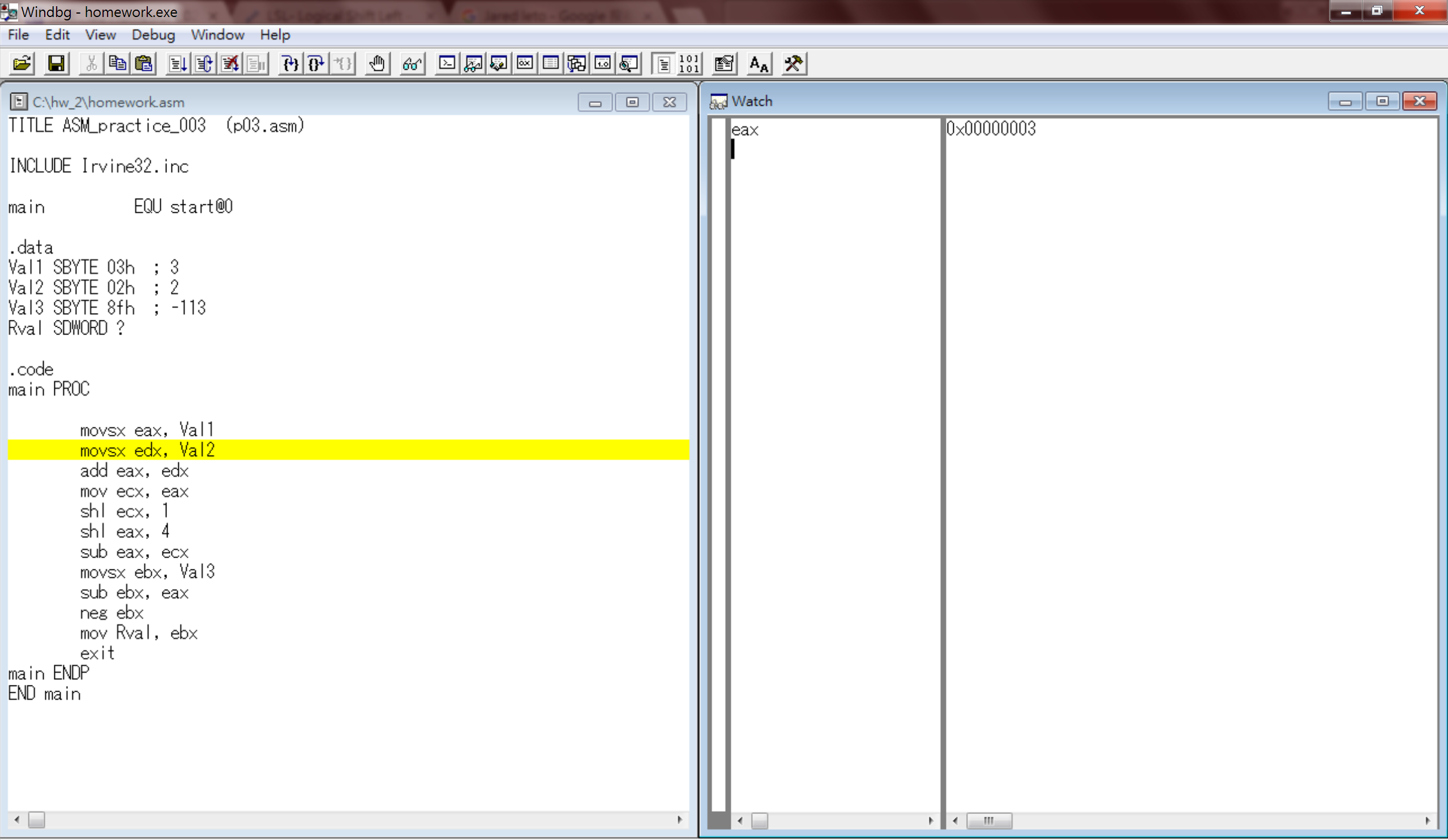
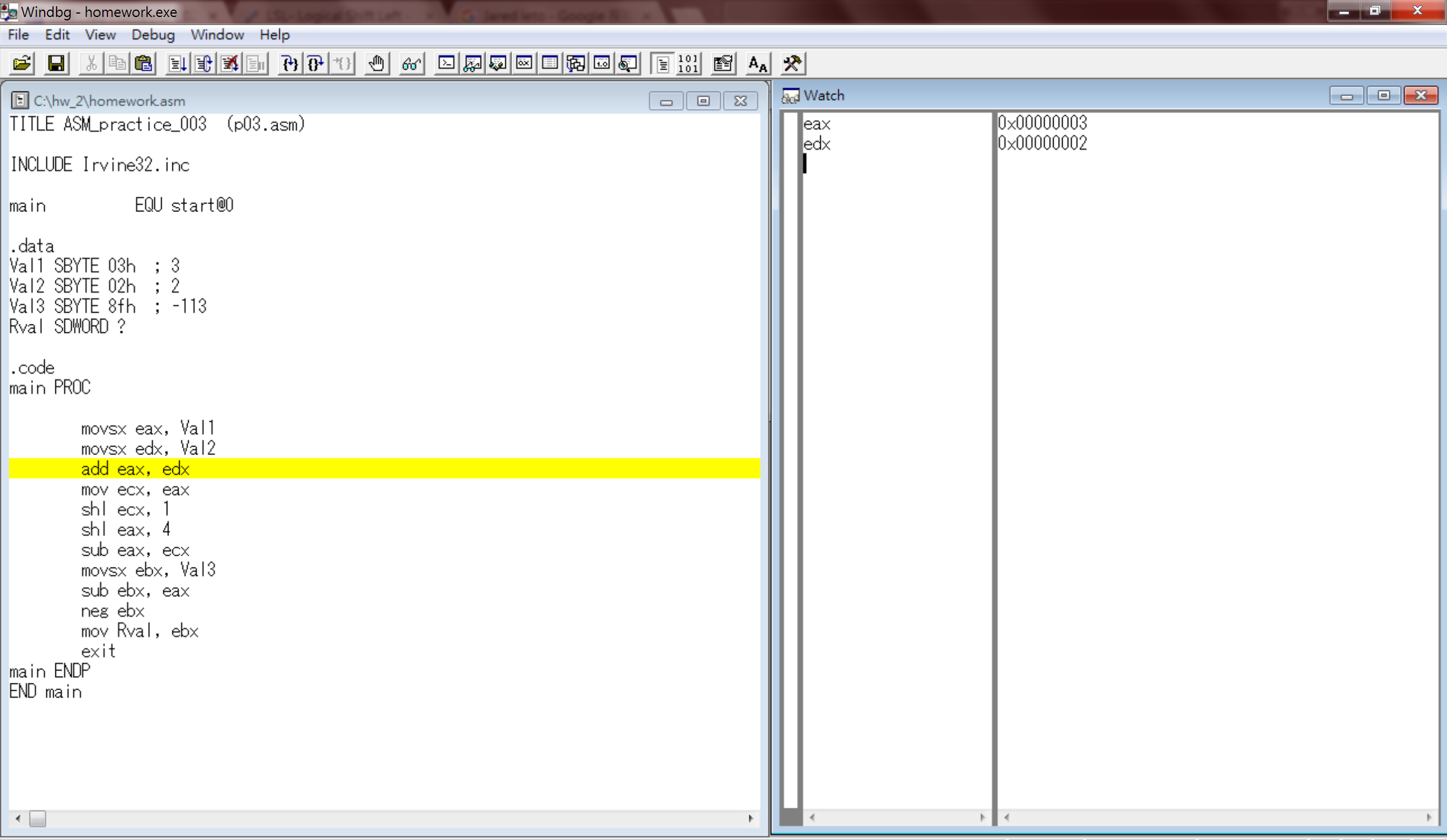
In-Class Exercise # 3

Group: 61

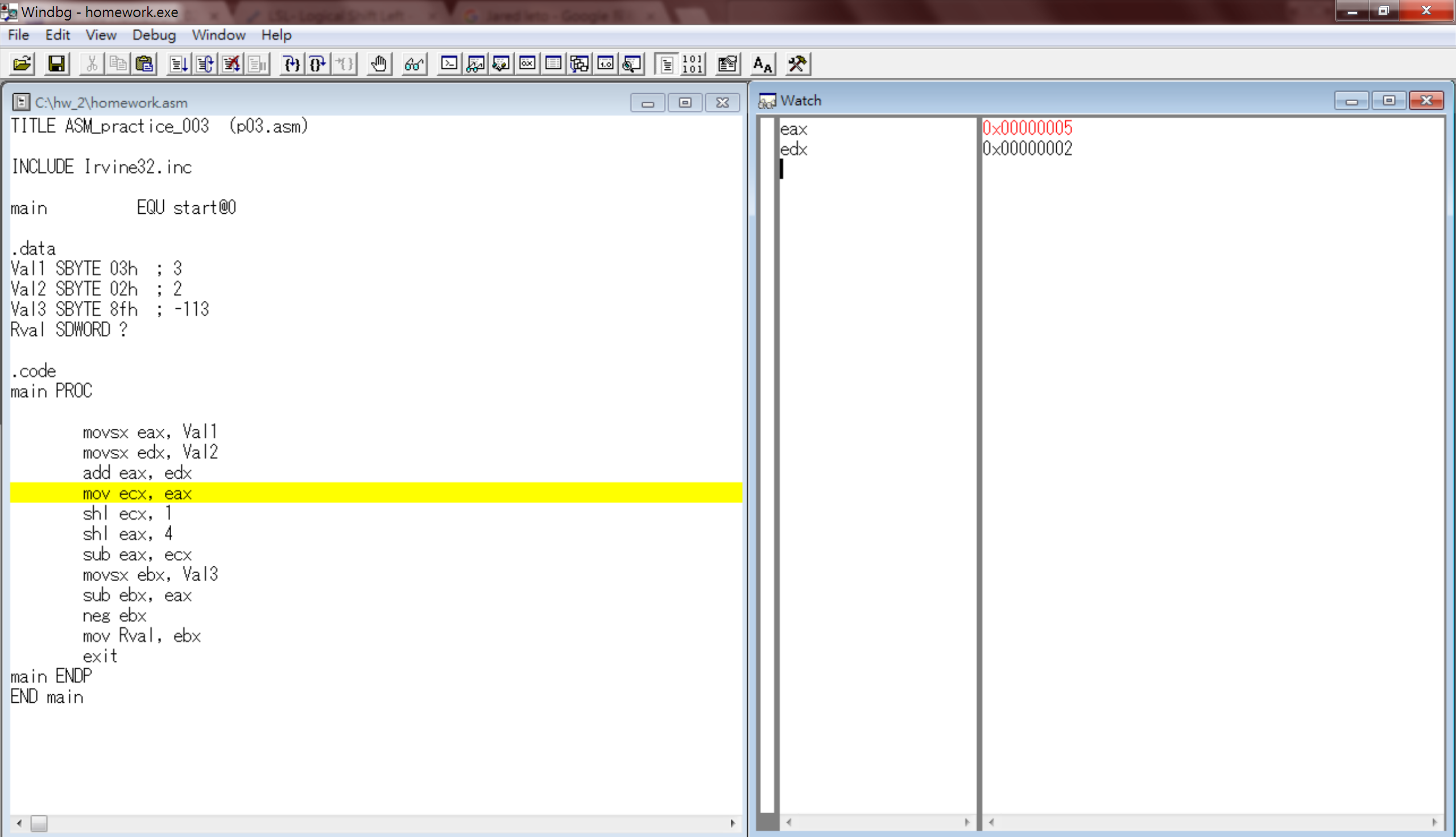
Name & ID: 陳宥涵(Joy) 102401053 趙慈暄(Bonnie) 104503002

Program execution flow

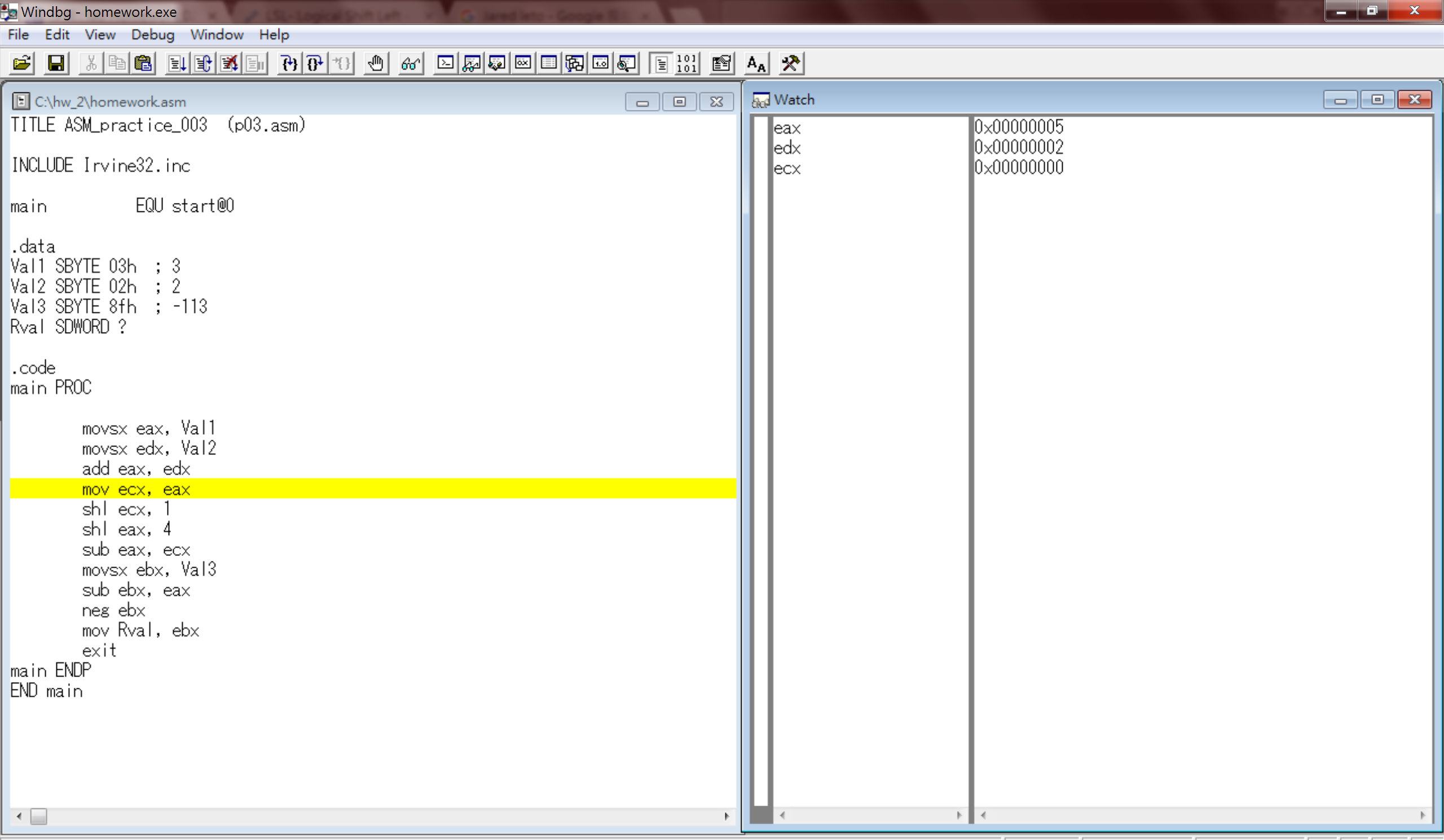
1. The following picture can see that after assigning Val1 to eax, eax became 0x00000003. Since Val1 is an SBYTE, which means it’s a 8 bits signed value, we have to use movsx to move the value to register eax.
2. This step is similar to the last step, since Val2 is an SBYTE, we have to use movsx to move Val2 to register edx.



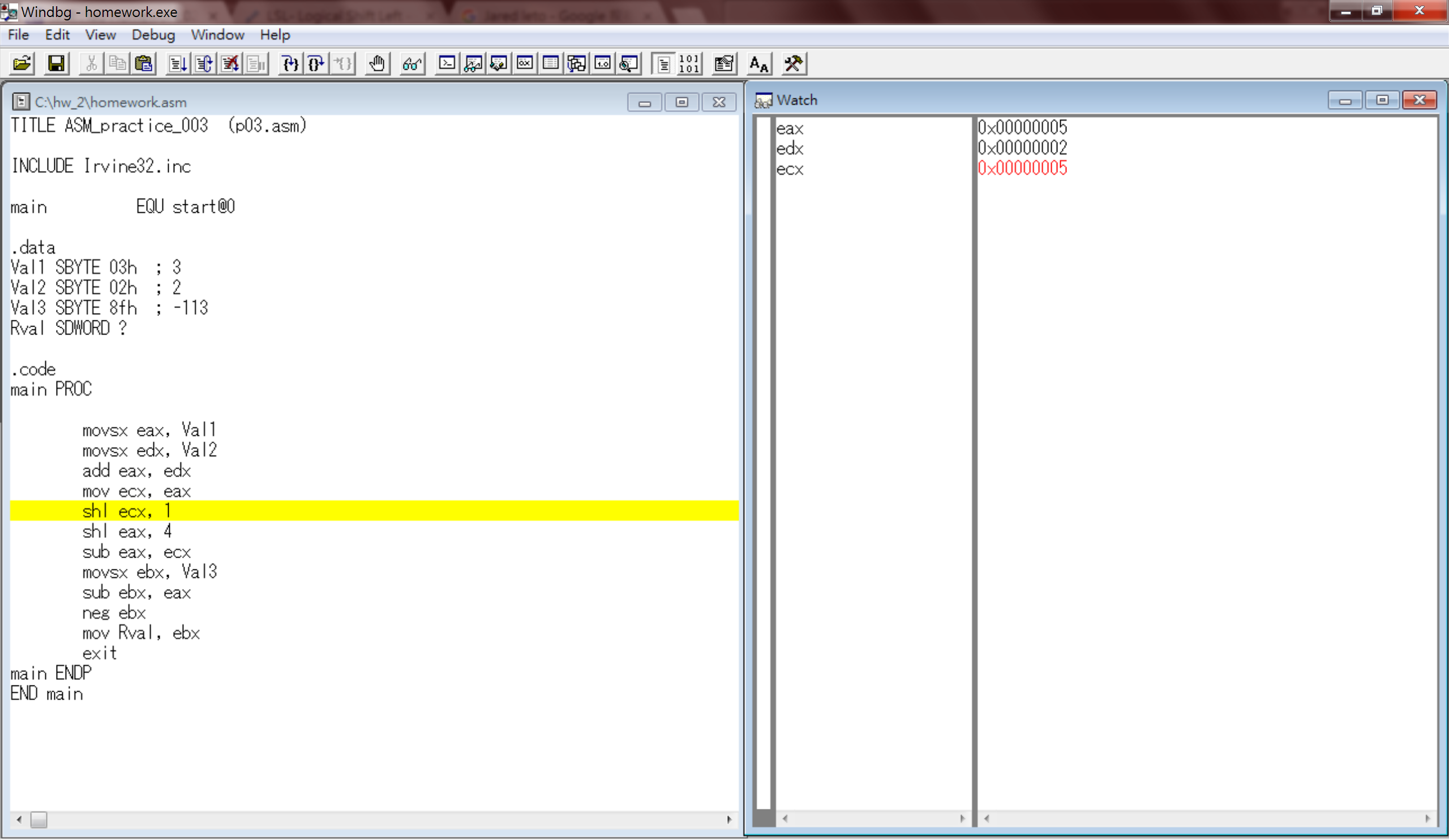
1. After adding edx to eax, the value of eax become 0x00000005. Since 3 +2 = 5, we know that the result is correct.



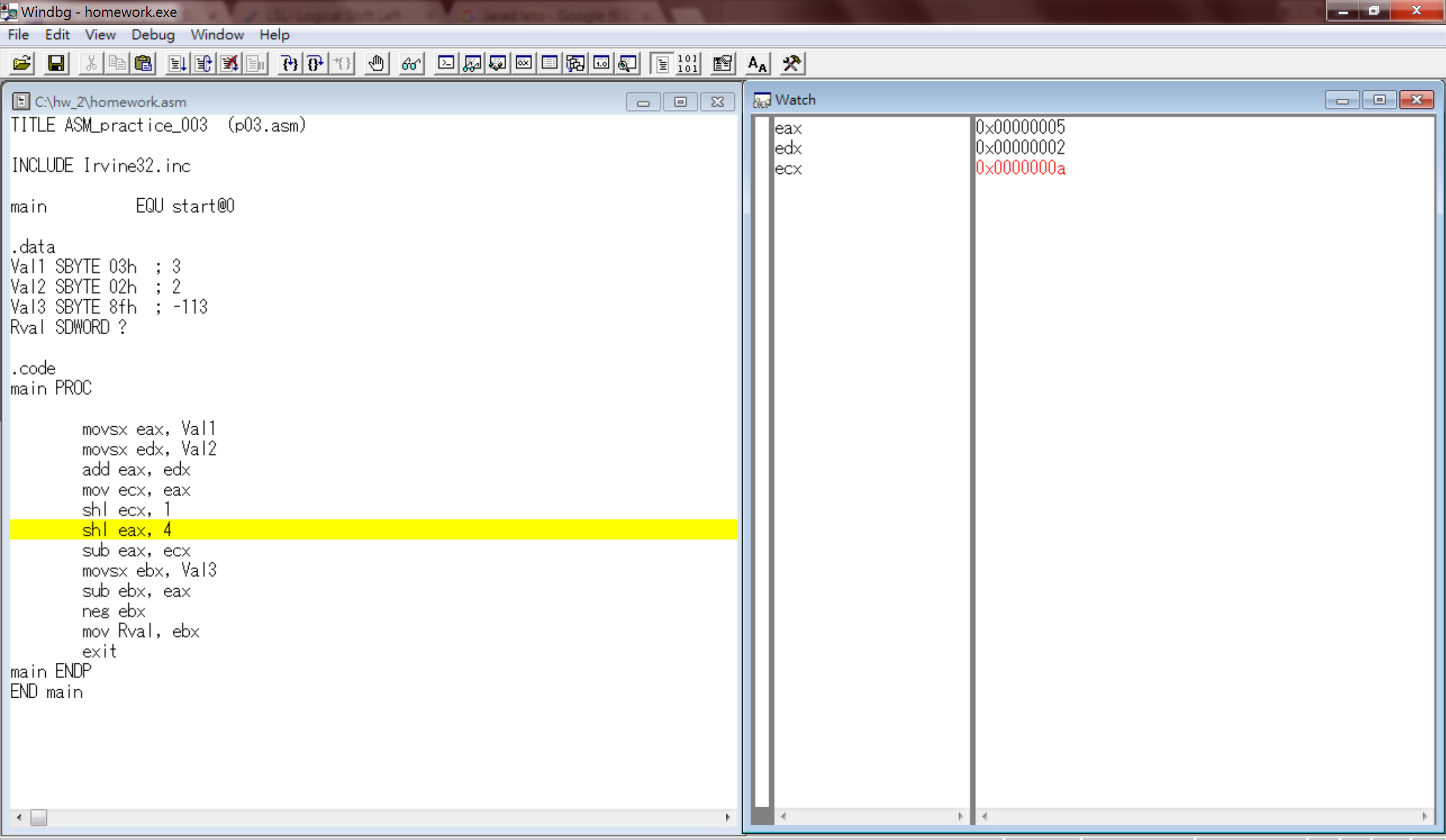
1. In order to multiply (Val1+Val2) by 14, we’ll have to first convert it to (Val1+Val2)\*16-(Val1+Val2)\*2, so we first copy the value of eax to ecx for further usage. In this step since I haven’t run the code to move eax to ecx, the initial value of ecx is 0x00000000.



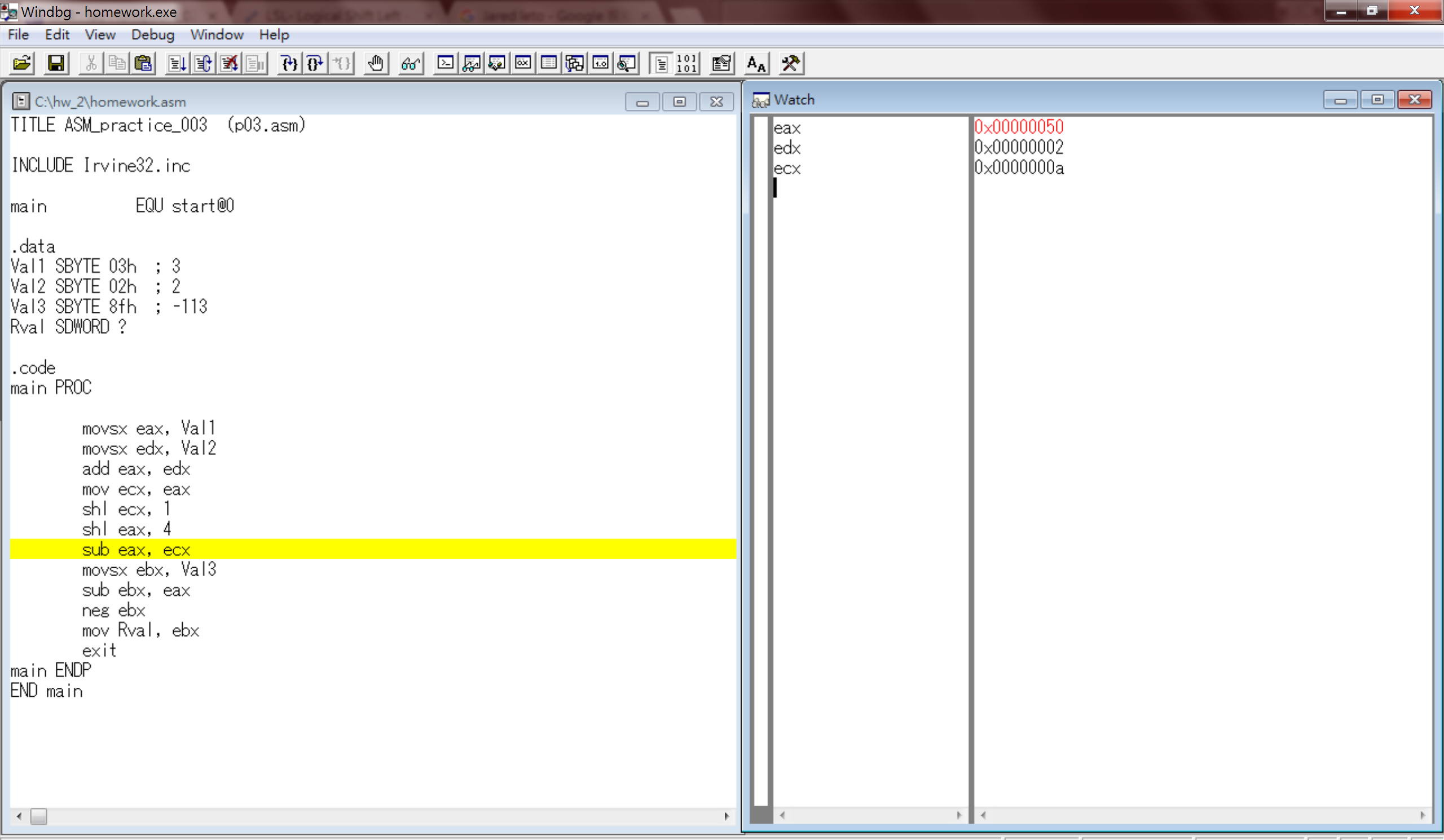
1. After I execute the code to move the value in register eax to ecx, we can now see that the value of register ecx has the same value of eax.



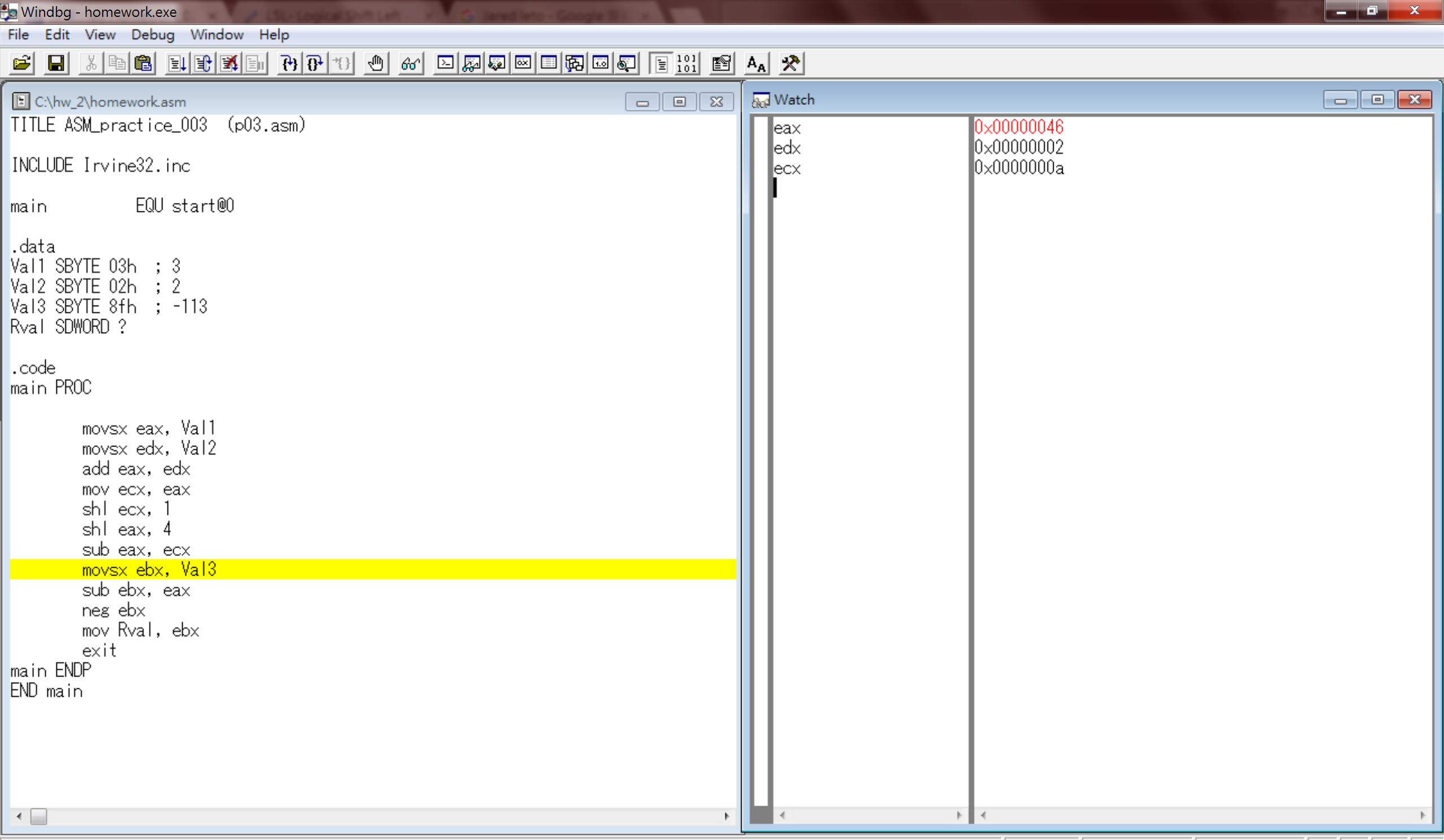
1. Since we’re going to first do the operation of (Val1+Val2)\*2, we left shift the value of ecx, which is (Val1+Val2). After 1 position left shifting we can get a value multiplied by 2 in register ecx.



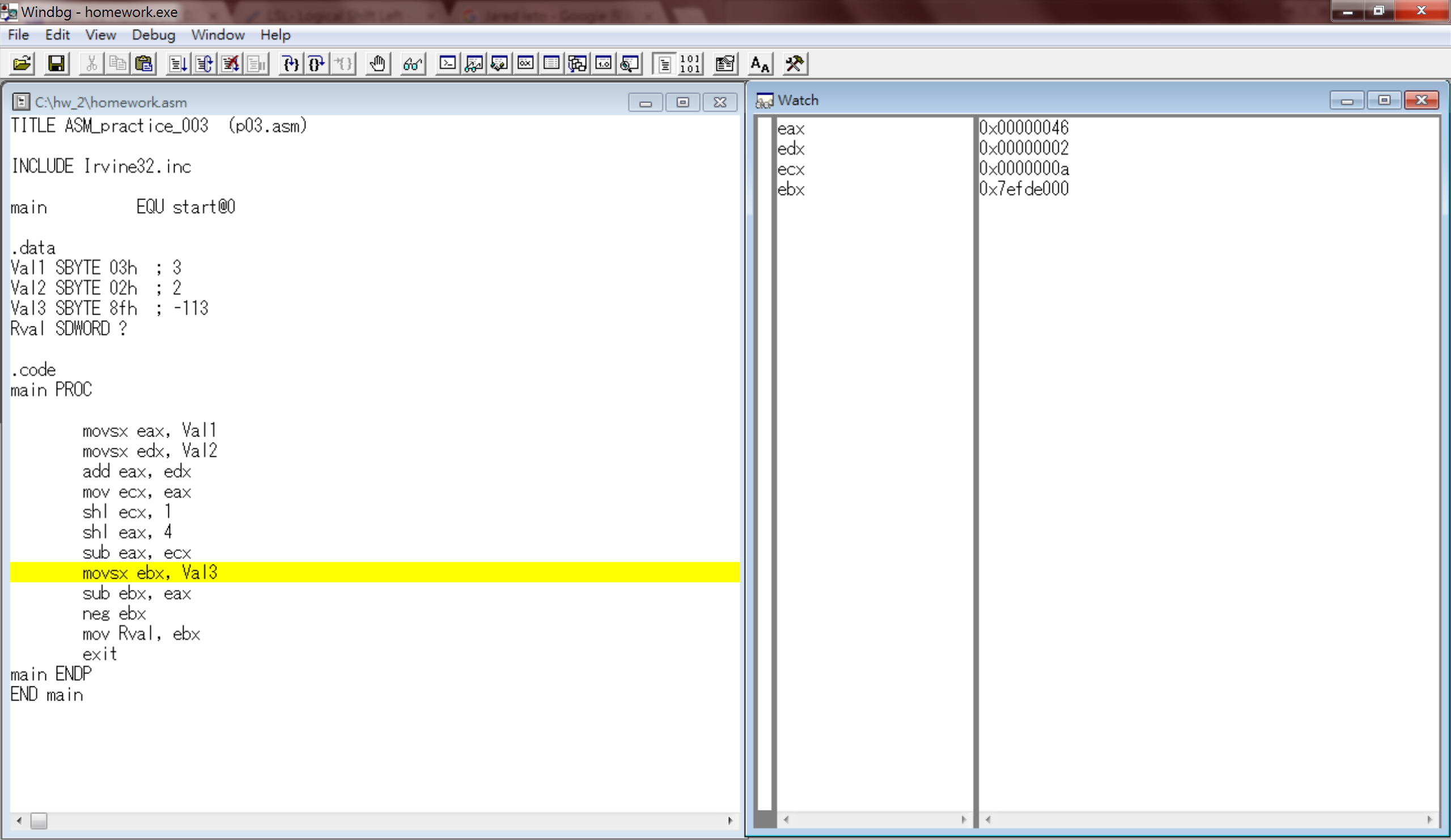
1. In this step we’ll do the operation: (Val1+Val2)\*16, which is to left shift for 4 position. Thus, after executing the code, we can see that the value in eax now became 50h, which previously was 5. We can double check that 16\*5=80, and 80d = 50h, so we got the correct number.



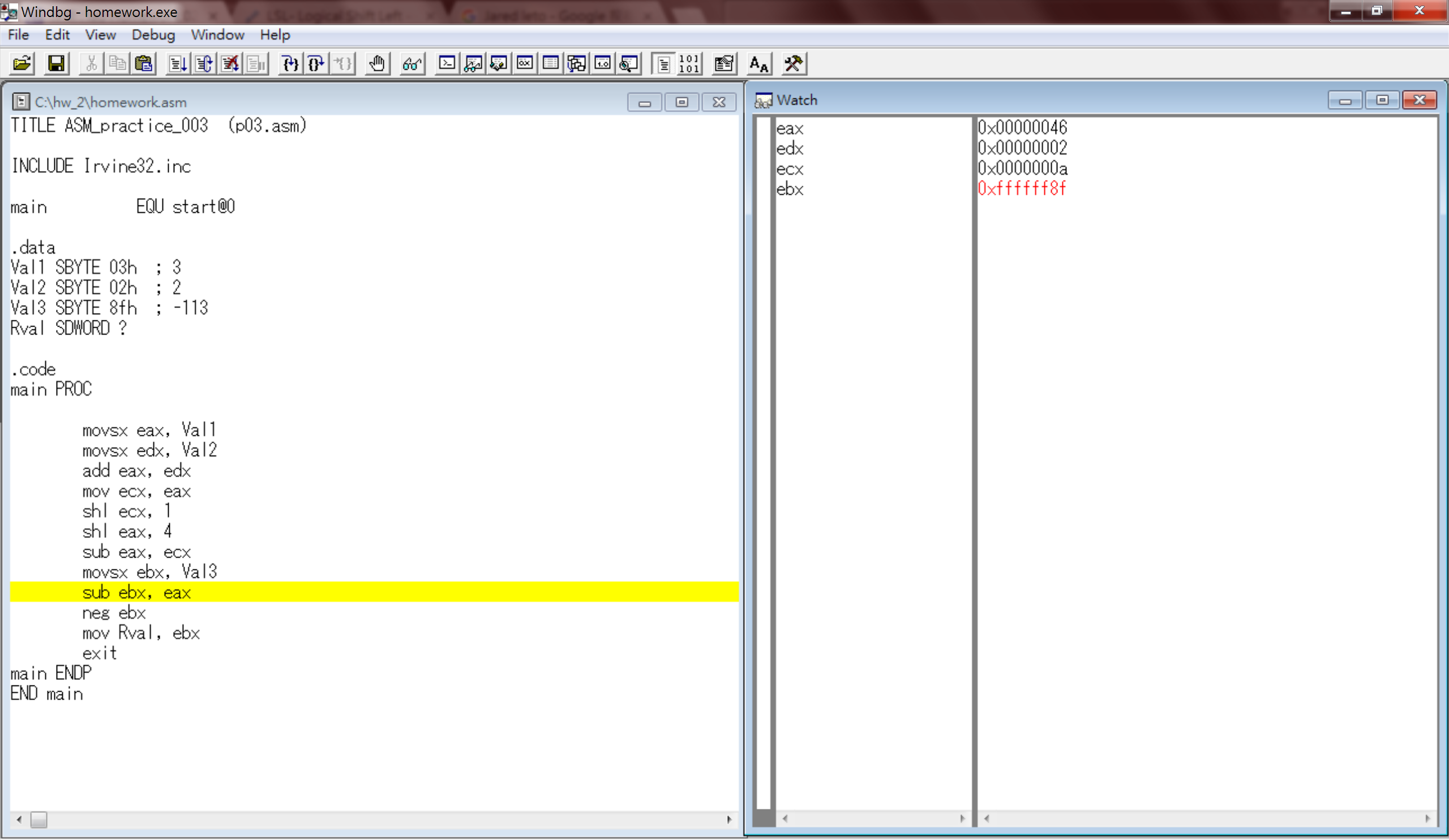
1. In order to get (Val1+Val2)\*14, we have to do (Val+Val2)\*16 - (Val+Val2)\*2, therefore, we subtract eax(Val+Val2)\*16 by ecx(Val+Val2)\*2. From the screenshot below, we can see that the value of eax became 46h, since 50h-0ah is 46h, we can know it’s correct.



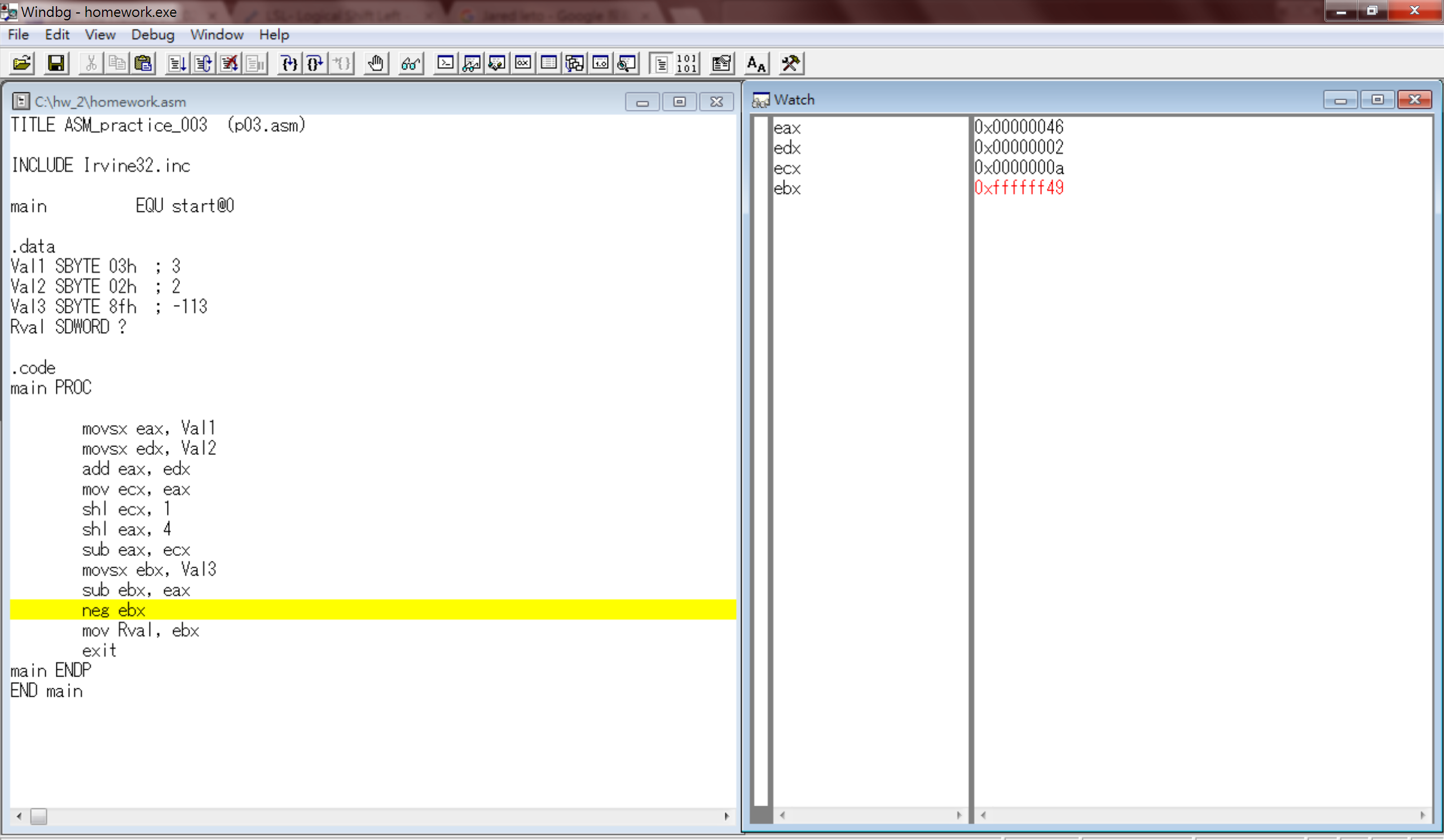
1. This step is just to take a look at the initial value of ebx, which is the register we’re about to use in the next step.

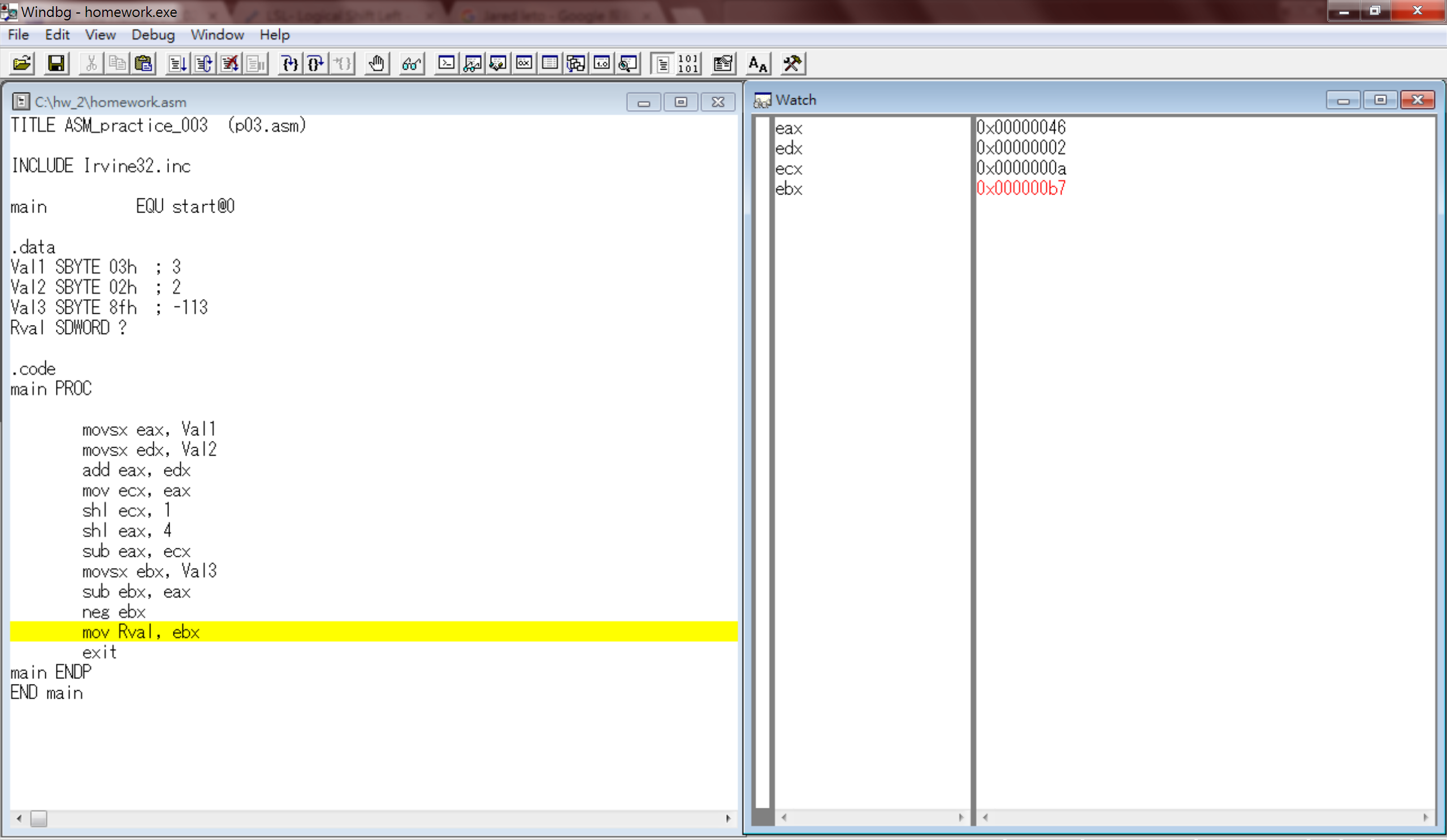


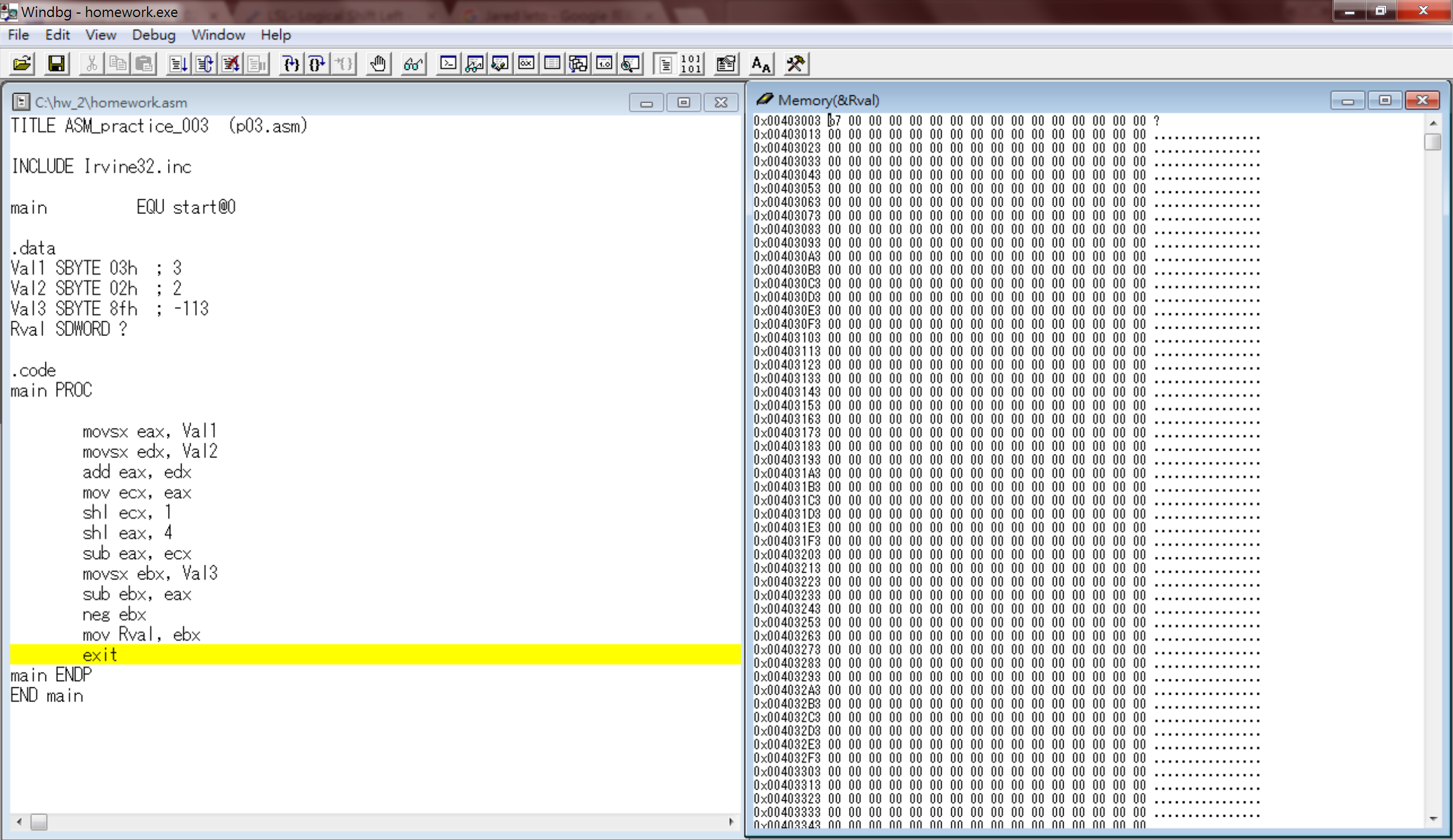
1. In this step, we can see the value of ebx after assigning Val3 to it, and since Val3 is a signed value, the empty bits in ebx will be filled with f to represent the negative sign.



1. In this step, we will conduct the operation of: Val3 – 14 \* (Val1 ＋ Val2), which is to subtract the value in eax by ebx. We can see that the value became 49h, which is correct since 8fh-46h = 49h.



1. In this step, we take the negation of ebx, which turns 0xffffff49 to 0x000000b7 by getting it’s 2’s complement.   
   
2. Last, after moving the value in register ebx, we can take a look at the variable Rval in the memory which is b7. It corresponds to the value of -(-113-14\*(3+2))=183d = b7h.



Reviews:

Today while I was coding, I found that I didn’t pay attention to some of the details like I had to use movsx instead of mov since the variable is a signed value, which cause some error. However, thanks to the hint from the TAs, I’m able to correct this mistake and also clarify misconceptions. In this lab session, I learned that when moving a signed value from memory to register, I had to use movsx, nevertheless, I don’t have to use movsx when moving from register to memory. Due to this assignment, I had a deeper understanding in this instructions and am more familiar with assembly code. After all, thanks to TAs help in the lab session!