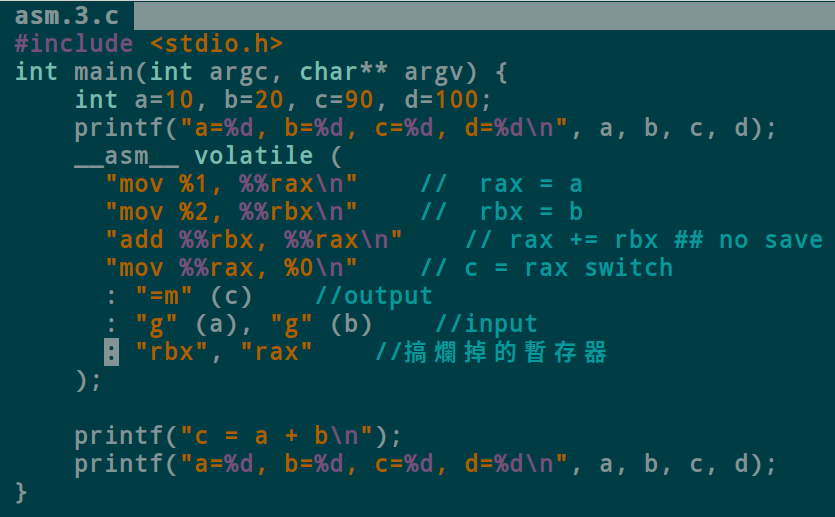
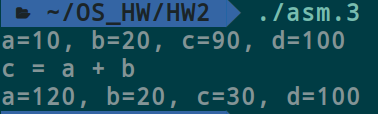
OS hw02.assembly 406410114 資工三 郭晏誠

1. **修正asm.3.c錯誤**

(1) 修改前





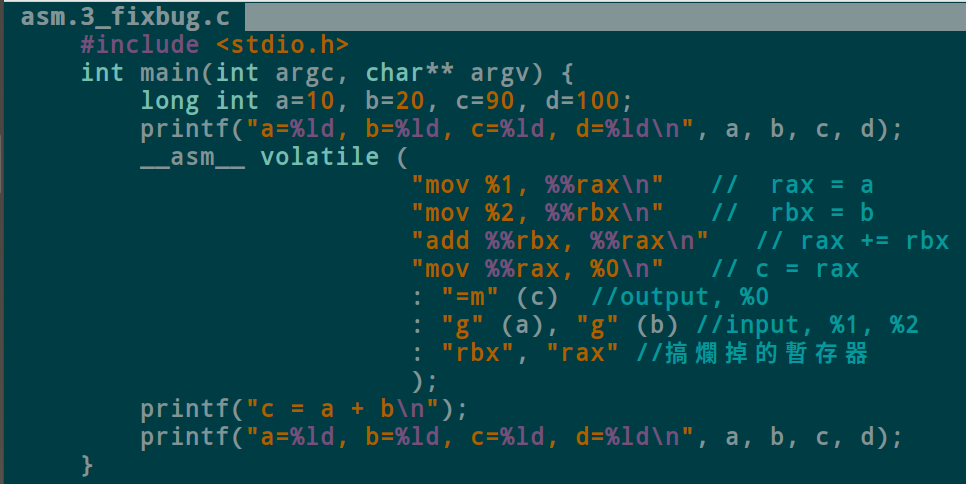
(2)修改後

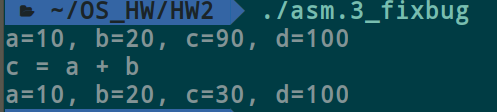
(i) 修正BUG：int改成long int

(ii) 原因： rax暫存器為64bit專用，用int可能會有錯誤，故把

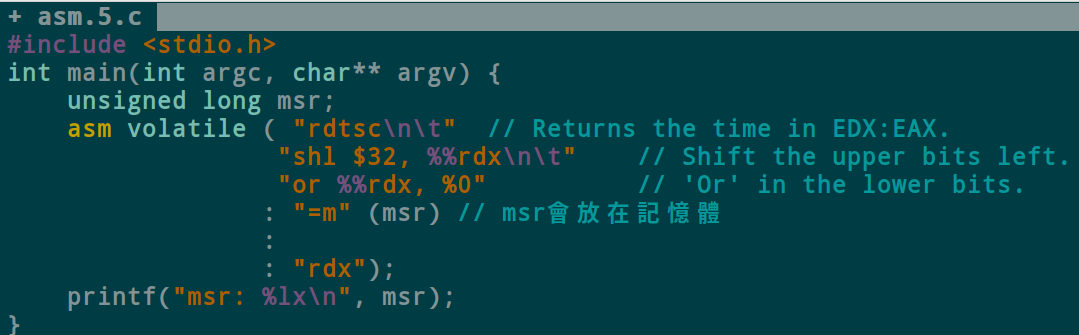
int改成long int

(iii) 其他解法：換成用eax暫存器來存int





1. **修正asm.5.c錯誤**
2. 修改前



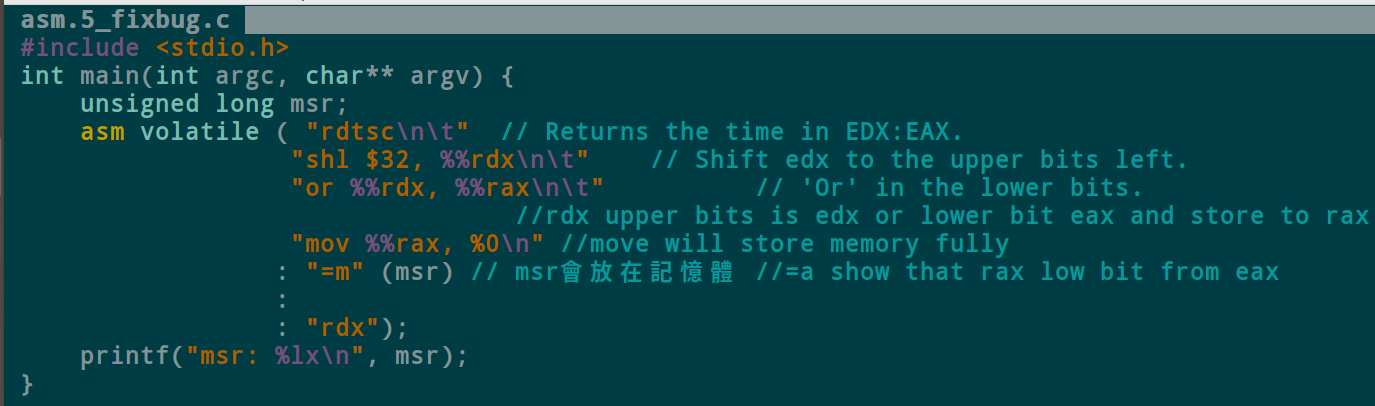


1. 修改後

(i) 修正BUG：先or存到rax暫存器再改用mov放進去記憶體

(ii) 原因： 用or直接寫入記憶體會導致儲存不完全，所以後面的位

元都是0，故改用mov存到記憶體





//rdtsc是用來取得Intel Pentium以上CPU的TSC值 (time stamp counters)，所經過的clock數, 其數值為64 bit, 傳回於EDX:EAX中

//附錄：Youtube筆記

C語言視高階的組合語言

Linux幾乎都用C，除了跟CPU相關的才會用組合語言寫

asm [volatile] //不要對下面的組合言做優化

( AssemblerTemplate //這部分就是組合語言

: OutputOperands // optional ，組語會輸出的變數

[ : InputOperands // optional，組語會讀取的變數

[ : Clobbers ] // optional ]，組合語言搞爛掉的暫存器的值)

//gcc會自動先存然後再用完後還原

//用指標存取記憶體比較麻煩

Asm.1.c (範例)

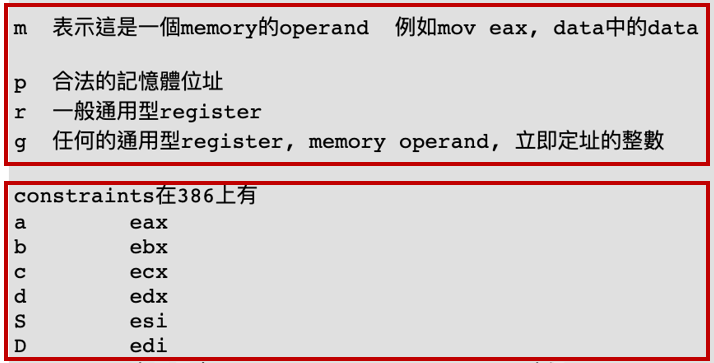
1. **\_\_asm\_\_** **volatile** ( //不要優化
2. "movl $100, %%eax\n"     *// eax = 100 movl是後面接的是多長*
3. "movl $200, %%ebx\n"     *// ebx = 100*
4. "addl %%ebx, %%eax\n"    *// eax += rbx b+a存到a*
5. "movl %%eax, %0\n"       *// b = rax*
6. : "=g" (b)              *//output, b的代號是"%0"*
7. : "g" (a), "g" (d)      *//input, a代號是"%1", da代號是"%2"*
8. : "ebx", "eax"          *//搞髒掉的暫存器，gcc會幫我們還原*
9. );

Asm.2.c (範例)

1. **\_\_asm\_\_** **volatile** (
2. "mov $100, %%rax\n" //不是movl ax暫存器 r代表

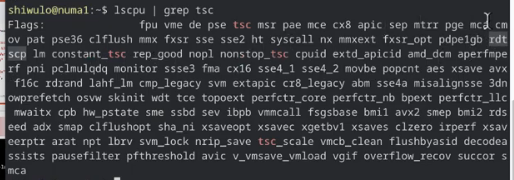
// 64位元 e是32位元

1. "mov $200, %%rbx\n"
2. "add %%rbx, %%rax\n"
3. "mov %%rax, %0\n"
4. : "=m" (b)    *//output*
5. : "g" (a), "g" (d) *//input*
6. : "rbx", "rax" *//搞爛掉的暫存器*



X86 運算沒有上指定operand，會有預設的

//lscpu | grep tsc 列出支援的TSC 原本是用來量Cycle 後來被用來量時間



//因為CPU會動態調整頻率 使得tsc會不準確

//所以有contant\_tsc 獨立於CPU的時脈

//nonstop\_tsc 提供較精準constant time

//avx2 向量指令集

//sse sse2 sse3 sse4\_1 see4\_2 向量指令集avx = see5, avx2 =see6 ( Intel ,AMD )

//avx2 512/8 = 64 一個運算最多處理64operand

//clzero 寫入zero來清除logical address

//rdrand CPU支援RANDOM

Asm.4.c (範例)

1. #include <stdio.h>
2. **int** main(**int** argc, **char**\*\* argv) {
3. **unsigned** **long** msr;
4. **asm** **volatile** ( "rdtsc\n\t"  *// Returns the time in EDX:EAX.*
5. "shl $32, %%rdx\n\t"    *// Shift the upper bits left.*
6. "or %%rdx, %0"          *// 'Or' in the lower bits.*
7. : "=a" (msr)            *//msr會放在rax暫存器 output*
8. : //input
9. : "rdx"); //搞爛的暫存器
10. printf("msr: %lx\n", msr);
11. }

Asm.5.c (作業)

1. #include <stdio.h>
2. **int** main(**int** argc, **char**\*\* argv) {
3. **unsigned** **long** msr;
4. **asm** **volatile** ( "rdtsc\n\t"  *// Returns the time in EDX:EAX.*
5. "shl $32, %%rdx\n\t"    *// Shift the upper bits left.*
6. "or %%rdx, %0"          *// 'Or' in the lower bits.*
7. : "=m" (msr)            *//msr會放在記憶體 一定要寫入記*

*//憶體修改上面*

1. :
2. : "rdx");
3. printf("msr: %lx\n", msr);
4. }

X86 記憶體裡面有隱藏的暫存器，讓程式碼變短，減少MEM使用空間