

# OS Homework 1

經濟三 407510046 黃郁雯

## 1. 修正第 32 頁錯誤：

```
#include <stdio.h>
int main(int argc, char** argv) {
    unsigned long msr;
    asm volatile ( "rdtsc\n\t" // Returns the time in EDX:EAX.
                  "shl $32, %%rdx\n\t" // Shift the upper bits left.
                  "or %%rdx, %%rax\n\t" // 'Or' in the lower bits.
                  "mov %%rax, %0\n"
                  : "=m" (msr) //msr會放在記憶體
                  :
                  : "rdx");
    printf("msr: %lx\n", msr);
}
```

因為照原本的 asm.5.c 的檔案，是直接放入記憶體計算，會產生錯誤。因此，修改成用兩個暫存器 rdx 和 rax 算完後，再將算好的值 mov 到記憶體中。

```
s407510046@lonux:~/sharedFolder/hw02.asm$ gcc asm.5.change.c -o asm.5.change
s407510046@lonux:~/sharedFolder/hw02.asm$ ./asm.5.change
msr: 216ca7e97dd1e4
s407510046@lonux:~/sharedFolder/hw02.asm$ ./asm.5
msr: 217fff8d525480
```

asm.5.change 為修改後的執行結果；asm.5 為修改前的執行結果。

## 2. 寫一支程式會自動觸發「除錯模式」(int3)：

```
1 #include <stdio.h>
2
3 int main(){
4     printf("os\n");
5     __asm("int3");
6     printf("gg\n");
7 }
8
```

上圖為一支簡易程式，我將第五行輸入\_\_asm("int3")，讓程式自動觸發除錯模式。

```

0x00007fffffff270 +0x0000: 0x0000555555554640 → <_libc_csu_init+0> push r15 ~ $rsp, $rbp
0x00007fffffff278 +0x0008: 0x00007ffff7a05b97 → <_libc_start_main+231> mov edi, eax
0x00007fffffff280 +0x0010: 0x0000000000000001
0x00007fffffff288 +0x0018: 0x00007fffffff3358 → 0x00007fffffff5c6 → "/home/normalUsers/s407510046/osdi/sharedFolder/hw0[...]"
0x00007fffffff290 +0x0020: 0x00000000100008000
0x00007fffffff298 +0x0028: 0x0000555555554615 → <main+0> push rbp
0x00007fffffff2a0 +0x0030: 0x0000000000000000
0x00007fffffff2a8 +0x0038: 0x6999d014794222c0

$rax : 0x0000555555554615 → <main+0> push rbp
$rbx : 0x0
$rcx : 0x0000555555554640 → <_libc_csu_init+0> push r15
$rdx : 0x00007fffffff3368 → 0x00007fffffff608 → "LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so[...]"
$rsp : 0x00007fffffff270 → 0x0000555555554640 → <_libc_csu_init+0> push r15
$rbp : 0x00007fffffff270 → 0x0000555555554640 → <_libc_csu_init+0> push r15
$rsi : 0x00007fffffff3358 → 0x00007fffffff5c6 → "/home/normalUsers/s407510046/osdi/sharedFolder/hw0[...]"
$rdi : 0x1
$rip : 0x0000555555554619 → <main+4> lea rdi, [rip+0xa4] # 0x5555555546c4
$r8 : 0x00007ffff7dd0d80 → 0x0000000000000000
$r9 : 0x00007ffff7dd0d80 → 0x0000000000000000
$r10 : 0x0
$r11 : 0x17
$r12 : 0x0000555555554530 → <start+0> xor ebp, ebp
$r13 : 0x00007fffffff3350 → 0x0000000000000001
$r14 : 0x0
$r15 : 0x0
$eflags: [ZERO carry PARITY adjust sign trap INTERRUPT direction overflow resume virtualx86 identification]
$cs: 0x0033 $ss: 0x002b $ds: 0x0000 $es: 0x0000 $fs: 0x0000 $gs: 0x0000

2
3 int main(){
4     printf("os\n");
5     _asm("int3");
6     printf("gg\n");

0x555555554610 <frame_dummy+0> jmp 0x555555554590
0x555555554615 <main+0> push rbp
0x555555554616 <main+1> mov rbp, rsp
0x555555554619 <main+4> lea rdi, [rip + 0xa4]
0x555555554620 <main+11> call 0x555555554510
0x555555554625 <main+16> int3

get> b
Note: breakpoint 1 also set at pc 0x555555554619.
Breakpoint 2 at 0x555555554619: file int3debug.c, line 4.
get>

```

可以從圈出來的地方看出：int3 會在前一行產生 breakpoint。