Homework8

Homework8

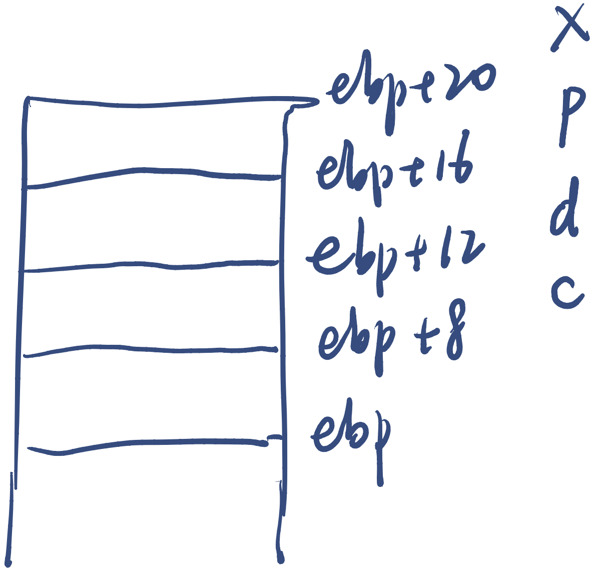
- 题目1
  - 一个C函数fun具有如下代码体：(参数从右向左入栈)

```
*p = d;
return x-c;
```

  - 执行这个函数体的IA32代码如下：

```
                        char
                   d
1 Movsbl    12(%ebp), %edx   ;较小的byte->dword, s表示符号填充， z表示0填充
                   p
2 Movl      16(%ebp), %eax
            d      p
3 Movl      %edx, (%eax)     short
                 c
4 Movswl    8(%ebp), %eax
                   x
5 Movl      20(%ebp), %edx
            c     x
6 Subl      %eax, %edx     }  return   x-c
7 Movl      %edx, %eax     }
```

写出函数fun的原型，给出参数p, d, x, c的类型和顺序。写出求解过程。



fun(short c, char d, int *p, int x)

Homework8

- 题目2
  - Suppose the initial value of %esp is 0x7FFFFFC4, initial value of %ebp is 0x7FFFFFF4.
  - The value stored in address 0x7FFFFFC0 is 0x120, value stored in address 0x7FFFFFC4 is 0x200, the value stored in address 0x7FFFFFF4 is 0x2710.
  - We have following x86 assembly code executed sequentially:
    - pushl %esp (instruction 1)
      movl %esp,%ebp (instruction 2)
      popl %ebp (instruction 3)
  - Question: After each instruction executed, what is the value of %esp and %ebp

(1) Instruction 1:	0x7FFFFFC0	0x7FFFFFF4
(2) Instruction 2:	0x7FFFFFC0	0x7FFFFFC0
(3) Instruction 3:	0x7FFFFFC4	0x7FFFFFC0
	%esp	%ebp

Homework8

- 题目3
  - 右边是C语言源代码文件func.c对应的汇编代码，请写出对应的C语言代码；
  - 画出Line 24执行前栈的状态，以及此时寄存器%edi, %esi, %edx, %ecx, %rsp的值；假设进入main函数前%rsp的值为0x8000420（代码中出现的局部变量，要标记在栈图中；图中标记内存地址）

```
1: .file    "func.c"
2:          .section        .rodata.str1.1,"aMS",@progbits,1
3: .LC0:
4:          .string "%d %d"
5:.LC1:
6:          .string "%d %d %d\n"
7:          .text
8:          .globl  main
9:          .type   main, @function
10:main:
11:.cfi_startproc
12:             subq    $24, %rsp
13:             leaq    8(%rsp), %rdx
14:             leaq    12(%rsp), %rsi
15:             movl    $.LC0, %edi
16:             movl    $0, %eax
17:             call    __isoc99_scanf
18:             movl    12(%rsp), %ecx
19:             movl    8(%rsp), %edx
20:             movl    %edx, %esi
21:             xorl    %ecx, %esi
22:             movl    $.LC1, %edi
23:             movl    $0, %eax
24:             call    printf
25:             movl    $0, %eax
26:             addq    $24, %rsp
27:             ret
```

↗ 18

rdx = rsp+8

rsi = rsp+12

edi = %d %d

eax = 0

ecx = [rsp+12] b

edx = [rsp+8] a

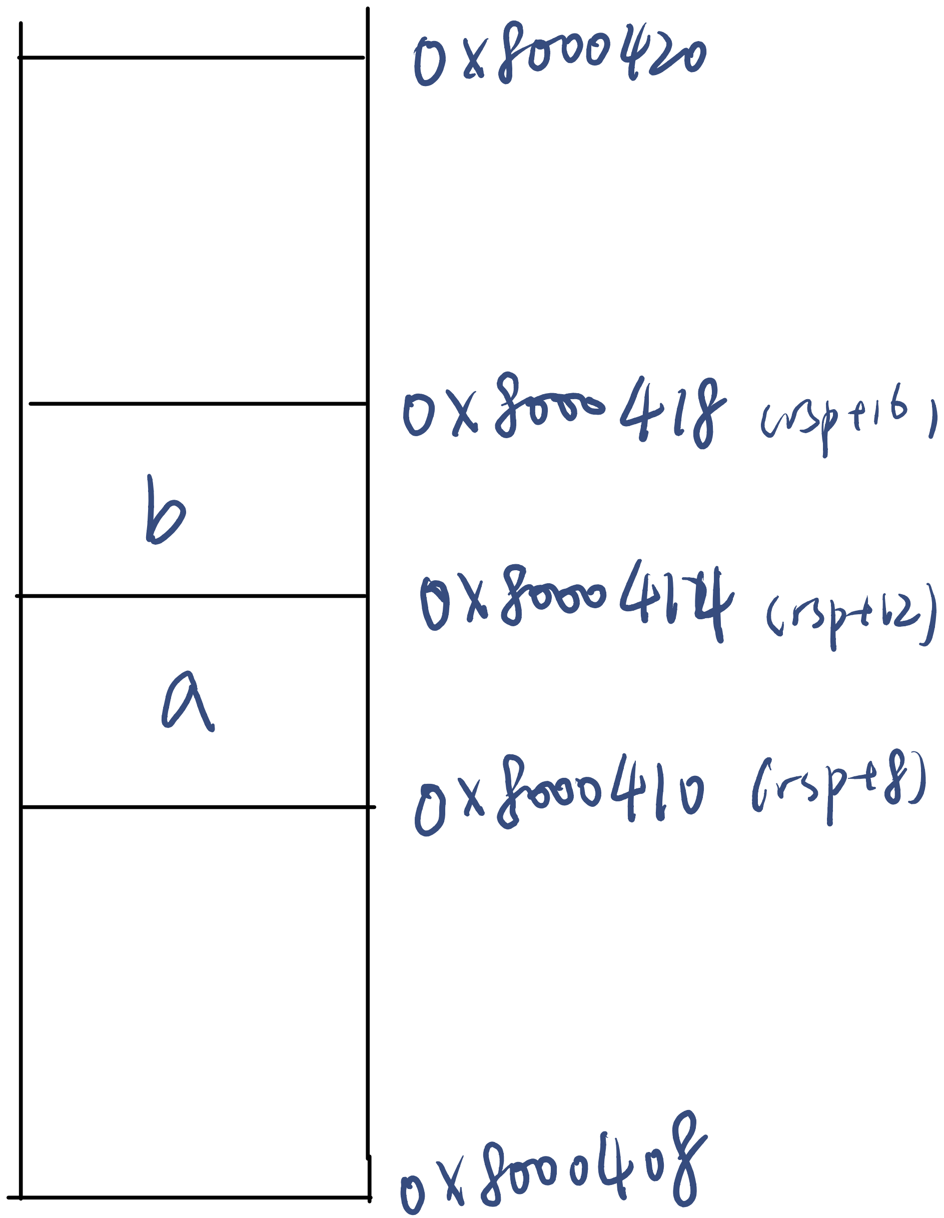
esi = edx

esi = ecx^esi

edi = %d %d %d

eax = 0

```
int a;
int b;
int c;
c = a^b;
```



%edi = $.LC1

%esi = a^b

%edx = a

%ecx = b

%rsp = 0x8000408