**\* 实验报告要求**

针对echo测试用例，在实验报告中，结合代码详细描述：

1. 注册监听键盘事件是怎么完成的？

我们对于echo.c中的函数进行分析：

文本

描述已自动生成

从端口读入一个字节的数据。

图形用户界面, 文本, 应用程序

描述已自动生成

注册监听事件：把Keyboard\_event\_handler和键盘输入的中断异常处理程序进行绑定。

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

键盘中断处理程序。

接下来我们来看执行的过程：

图形用户界面, 文本, 应用程序

描述已自动生成

执行int 0x80, 于是我们按照提示找到kernel/src/syscall/do\_syscall.c：

文本

中度可信度描述已自动生成

由eax=0，知道进一步调用add\_irq\_handle，注册监听事件，将Keyboard\_event\_handler和键盘输入的中断异常处理程序进行绑定。

void add\_irq\_handle(int irq, void (\*func)(void) )

{

assert(irq < NR\_HARD\_INTR);

assert(handle\_count <= NR\_IRQ\_HANDLE);

struct IRQ\_t \*ptr;

ptr = &handle\_pool[handle\_count ++]; /\* get a free handler \*/

ptr->routine = func;

ptr->next = handles[irq]; /\* insert into the linked list \*/

handles[irq] = ptr;

}

插入到空的处理函数位置。

1. 从键盘按下一个键到控制台输出对应的字符，系统的执行过程是什么？如果涉及与之前报告重复的内容，简单引用之前的内容即可。

此时，我们已经注册了键盘监听事件，将键盘中断与键盘中断处理程序Keyboard\_event\_handler绑定。现在，我们按下一个键，操作系统检测到中断（具体详见pa4-1），调用中断处理程序keyboard\_event\_handler, 首先执行指令in (in\_byte函数)，in指令中调用pio\_read，把读取的结果存储在eax中，返回搭配in\_byte函数输出。传入到translate\_key中将键码转化为对应的ascii码，再进一步调用printc, 通过printc调用writec, 而writec:

"int $0x80" : : "a"(SYS\_write), "b"(fd), "c"(&c), "d"(1)

由int $0x80 进一步调用sys\_write, sys\_write 调用fs\_write，fs\_write进一步调用，一直调用到out\_byte函数，通过out指令进行输出。

等待下一次中断到来。