PA4-2 Report

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?

`/testcase/src/echo.c`的`main()`函数中将监听键盘事件的处理函数`keyboard_event_handler()`注册至`1`号中断：

```
add_irq_handler(1, keyboard_event_handler);
```

`add_irq_handler`函数由内联汇编实现，用`int $80`中断进行系统调用，传入0号系统调用，注册中断号`irq`，待注册中断处理函数的指针`handler`.

```
asm volatile("int $0x80"
                 :
                 : "a"(0), "b"(irq), "c"(handler));
```

程序执行到这里时，将由`irq_handle`响应这一中断, 并交给`do_syscall`函数处理：

```
if (irq == 0x80)
    {
        do_syscall(tf);
    }
```

`do_syscall`函数中对0号调用的处理为将传入的中断号及处理函数注册。

```
case 0:
        cli();
        add_irq_handle(tf->ebx, (void *)tf->ecx);
        sti();
```

```
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;
}
```

2. 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。

①等待键盘中断信号

```
while (1)
        asm volatile("hlt");
```

当中断信号到来CPU才会做出响应

②键盘有按键按下，键盘将响应键码转换为扫描码，并通过`i8259`中断控制器产生中断信号`KEYBOARD_IRQ`.

```
void keyboard_down(uint32_t sym)
{
    // put the scan code into the buffer
    scan_code_buf = sym2scancode[sym >> 8][sym & 0xff];
    // issue an iterrupt
    i8259_raise_intr(KEYBOARD_IRQ);
    // maybe the kernel will be interested and come to read on the data port
}
```

中断号是1.

③中断处理

cpu收到中断信号后，在执行下一条指令前响应中断：

```
if (cpu.intr && cpu.eflags.IF)
    {
        is_nemu_hlt = false;
        // get interrupt number
        uint8_t intr_no = i8259_query_intr_no(); // get interrupt number
        assert(intr_no != I8259_NO_INTR);
        i8259_ack_intr();   // tell the PIC interrupt info received
        raise_intr(intr_no); // raise intrrupt to turn into kernel handler
    }
```

之后将由之前注册的1号中断处理函数处理该中断，其过程在4-1中已有过说明，这里不再赘述。

具体的处理函数`keyboard_event_handler`如下：

```
void keyboard_event_handler()
{

    uint8_t key_pressed = in_byte(0x60);

    // translate scan code to ASCII
    char c = translate_key(key_pressed);
    if (c > 0)
    {
        // can you now fully understand Fig. 8.3 on pg. 317 of the text book?
        printc(c);
    }
}
```

从0x60号端口读出按下键的键码，将之转换为ASCII码，通过`printc`函数将字符打印. 将由`writec()`函数调用系统调用在终端打印该字符.

④中断处理完毕，回到①.