

第四次实验问题清单

By 张洪胤

问题一：进程是什么

进程是计算机中的程序关于某个数据集合上的一次运行活动，
是系统进行资源分配和调度的基本单位。

从宏观来看，它有自己的目标（功能），同时又能受控于进程调度模块；

从微观来看，它可以利用系统的资源，有自己的代码和数据，同时拥有自己的堆栈，进程需要被调度。

问题二：进程表是什么

进程表是存储进程状态信息的数据结构。

进程表是进程存在的**唯一标识**，是操作系统用来**记录和刻画进程状态及环境信息的数据结构**，是进程动态特征的汇集，也是操作系统掌握进程的唯一资料结构和管理进程的主要依据。

问题三：进程栈是什么

进程运行时自身的堆栈。

问题四：当寄存器的值已经被保存到进程表内，esp应指向何处来避免破坏进程表的值

进程运行时，**esp**指向堆栈中的某个位置。寄存器的值刚刚被保存到进程表内，**esp**是指向进程表某个位置的。如果接下来进行任何的堆栈操作，都会破坏掉进程表的值。

为解决这个问题，使用**内核栈**，让**esp**指向内核栈。

问题五：TTY是什么

Teletype的缩写。终端是一种字符型设备,它有多种类型,通常使用TTY来简称各种类型的终端设备。不同TTY对应的输入设备是同一个键盘。

问题六：不同的TTY为什么输出不同的画面在同一个显示器上

不同TTY各有一个**CONSOLE**，各个**CONSOLE**公用同一块显存。

虽然不同的TTY 对应的输入设备是同一个键盘，但输出却好比是在不同的显示器上，因为不同的TTY对应的屏幕画面可能是迥然不同的。实际上，我们当然是在使用同一个显示器，画面的不同只不过是显示了显存的不同位置罢了。

问题七：解释TTY任务执行的过程

在TTY任务中执行一个循环，这个循环将轮询每一个TTY，处理它的事件，包括从键盘缓冲区读取数据、显示字符等内容。

轮询到每一个TTY 时：

1. 处理输入：查看其是否为**当前TTY**。只有当某个TTY对应的控制台是当前控制台时，它才可以读取键盘缓冲区。
2. 处理输出：如果有要显示的内容则显示它。

问题八：TTY结构体中大概包括哪些内容

```
#define TTY_IN_BYTES 256 /* tty input queue size */

struct s_console;

/* TTY */
// tty 的结构体，代表 TTY
typedef struct s_tty
{
    u32  in_buf[TTY_IN_BYTES]; /* TTY 输入缓冲区 */
    u32* p_inbuf_head; /* 指向缓冲区中下一个空闲位置 */
    u32* p_inbuf_tail; /* 指向键盘任务应处理的键值 */
    int  inbuf_count; /* 缓冲区中已经填充了多少 */

    // 持有一个 console
    struct s_console * p_console;
}TTY;
```

问题九：Console 结构体中大概包括哪些内容

```
// Ctrl 缓冲区大小
#define CTRL_LENGTH 500

/* CONSOLE */
// console 结构体
typedef struct s_console
{
    unsigned int  current_start_addr; /* 当前显示到了什么位置 */
    unsigned int  original_addr; /* 当前控制台对应显存位置 */
    unsigned int  v_mem_limit; /* 当前控制台占的显存大小 */
    unsigned int  cursor; /* 当前光标位置 */
}CONSOLE;
```