

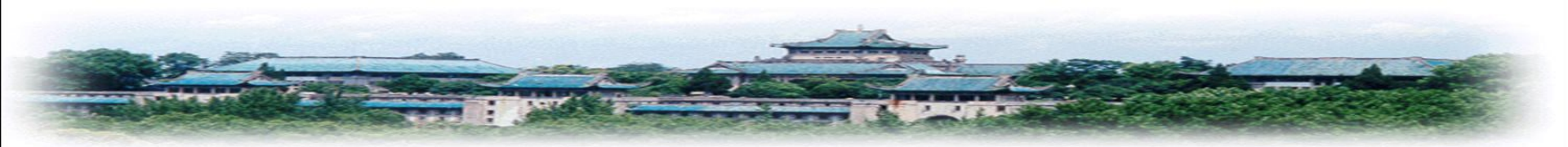
操作系统设计及实践

《操作系统原理》配套实验

信安系操作系统课程组

2024年11月



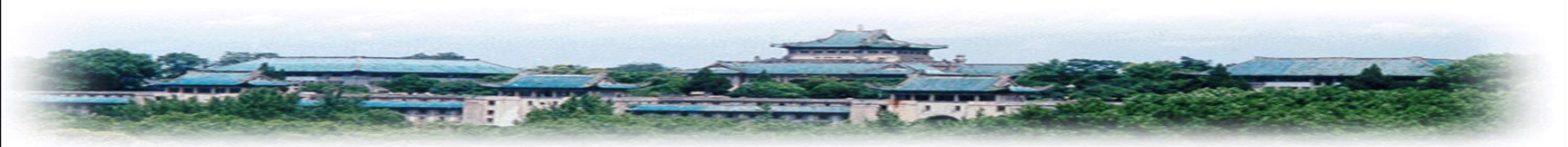


操作系统设计实验系列（八）

进程（二）：多进程与进程调度



武汉大学



一、实验目标

- 多进程的实现机理与进程调度
- 对应章节：第六章6.4、6.5、6.6





二、本次实验基本内容

1. 多进程问题，如何扩展单进程到多进程，如何扩展中断支持多进程？
2. 如何实现系统调用
3. 进程调度问题，弄清楚实现调度的基本思路





三、本次实验要解决的问题

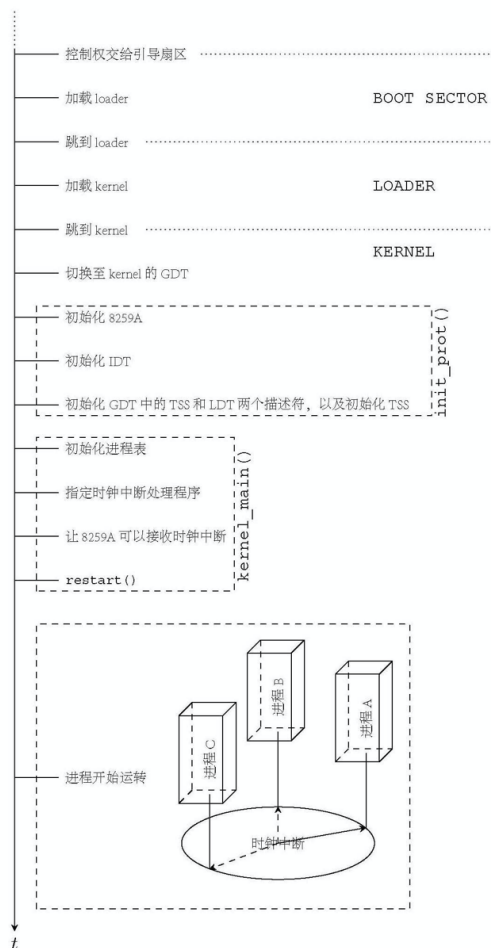
1. 在单进程的基础上扩展实现多进程要考虑哪些问题？
2. 画出以下关键技术的流程图：
 - 初始化多进程控制块的过程、扩展初始化LDT和TSS
3. 如何修改时钟中断来支持多进程管理，画出新的流程图。
4. 系统调用的基本框架是如何的，应该包含哪些基本功能，画出流程图。
5. 如何操控可编程计数器？
6. 进程调度的框架是怎样的？优先级调度如何实现？
7. 动手做：修改例子程序的调度算法，模拟实现一个多级反馈队列调度算法，并用其尝试调度多个任务。注意，抢占问题，注意时间片问题。鼓励使用其他更复杂的调度算法，如CFS等。
8. 思考题：从用户态进程读和写内核段的数据，看能否成功。



四、需要回顾了解的一些知识

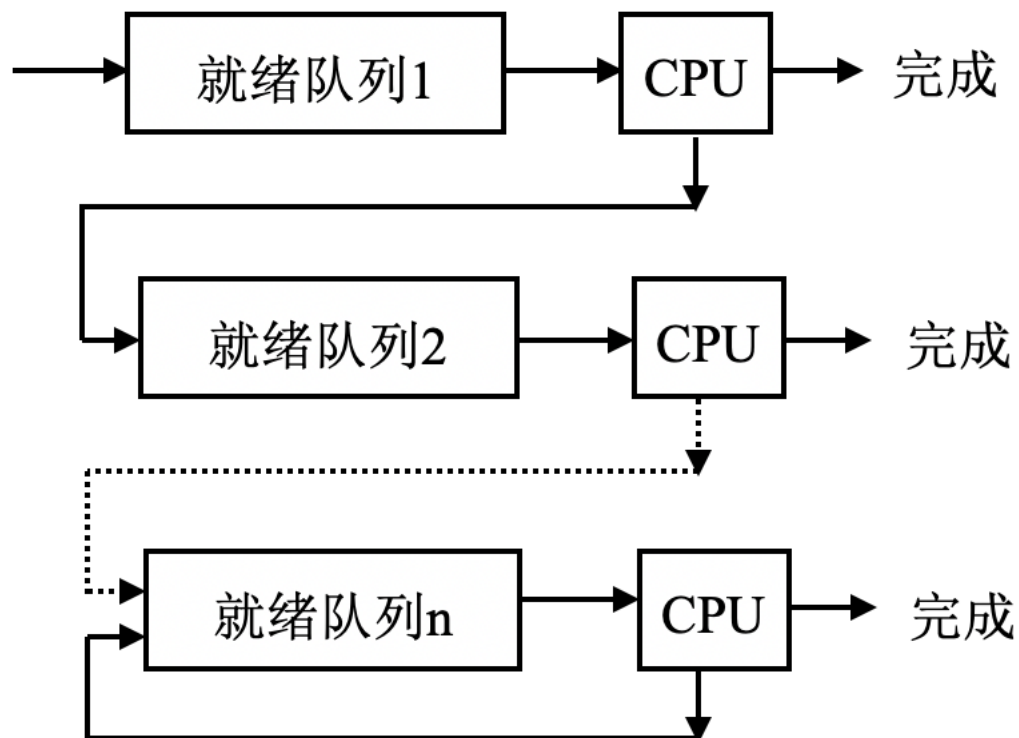
1. 多进程调度框架


- 如何实现引导扇区
- Loader加载
- Kernel加载
- 转换控制权给Kernel
- 初始化中断控制器
- 初始化进程管理模块
- 多进程调度



四、需要回顾了解的一些知识

2. 多级反馈队列





四、需要回顾了解的一些知识

3. 系统调用的设计原理


初始化中断门，大家可以参考/I/目录下代码

```
init_idt_desc(INT_VECTOR_SYS_CALL, DA_386IGate,  
              sys_call, PRIVILEGE_USER);
```

断对应的函数sys_call中调用入口参数对应的函数

```
sys_call:  
    call    save  
    sti  
    call    [sys_call_table + eax * 4]  
    mov     [esi + EAXREG - P_STACKBASE], eax  
    cli  
    ret
```





四、需要回顾了解的一些知识

3. 系统调用的设计原理

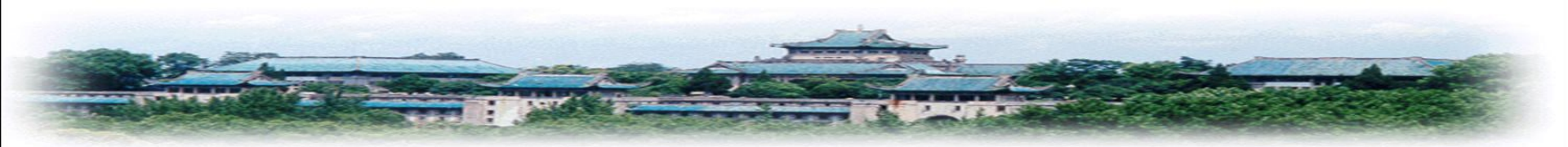
sys_call_table定义, global.c

```
PUBLIC system_call sys_call_table[NR_SYS_CALL] =  
    {sys_get_ticks};
```

get_ticks的系统调用

```
get_ticks:  
    mov eax, _NR_get_ticks  
    int INT_VECTOR_SYS_CALL  
    ret
```





谢 谢！



武汉大学