project3 design review

38组 王苑铮 姚子薇

What is the workflow for handling interrupt?

- (handle_int: entry.S)
- 1、关中断 enter_critical
- 2、保存中断上下文：SAVE_CONTEXT(USER)，保存进程的用户态信息。
- 3、判断中断类型（这次只考虑时钟中断）
- 检查CP0 IP7 bit:
- 如果是1: Clock interrupt！继续时钟中断处理
- 如果是0: 跳转到中断返回处理

What do you do in the clock interrupt handler?

- //1、关中断 enter_critical
- 1、增加计数器 time_elasped+1
- 2、检查nested_count TEST_NESTED_COUNT
- 如果是1，继续进行
- 如果是0，将它修改为1后继续进行
- 3、将当前进程加入到就绪队列中put_current_running()
- 4、调度新任务 scheduler_entry()
- SAVE_CONTEXT(KERNEL) & SAVE_CONTEXT(USER) -> scheduler
- -> 清中断-> 开中断
- ->RESTORE_CONTEXT (线程KERNEL 进程USER且nested_count改0)

中断返回处理

- 1、清中断 clz函数， 找到CAUSE寄存器 IP 域中不为 0的最高位，清零。若是时钟中断，即IP7 bit位清零。
- 2、开中断 LEAVE_CRITICAL
- 3、恢复上下文
- RESTORE_CONTEXT
- 线程KERNEL
- 进程USER nested_count改0

How do you implement blocking sleep?

- 1. do_sleep(): 当前进程阻塞睡眠，调用下一个需要运行的。
- 由线程主动调用。

- 因为要进入临界区，所以关中断 enter_critical();
- 计算deadline(time_elapsed*1000 + milliseconds) (单位);
- 更改状态为SLEEPING;
- 插入阻塞睡眠队列sleep_wait_queue;
- scheduler_entry().

How do you implement blocking sleep?

- 2. check_sleeping(): 唤醒那些当前时间已经达到所需运行的 task;
- 用while依次检查sleep_wait_queue中的每个任务;
- 凡是睡眠时间达到(sleeping->deadline <= time_elapsed*1000)，就将状态改为READY，插入ready队列里.

- 3. put_curring_running(): 保存当前进程 。
- put_current 把当前任务改成READY;
- 将当前任务放到ready队列里.

When to wake up the task?

- 已写好的scheduler函数里有check_sleepping(), 所以调用scheduler_entry()，都会检查唤醒所有睡眠时间已到的任务.
- 调用scheduler_entry()的函数有：
- 时钟中断;
- do_yield, do_exit, do_block, do_sleep;

- put_current_running()里不需要写check_sleepping().

How do you implement the priority based scheduler?

pcb[0]	pcb[1]	pcb[2]	pcb[3]	pcb[4]
W0	W0+W1	W0+W1+W2	W0+W1+W2+W3	W0+W1+W2+W3+W4

用随机数函数模W=W0+W1+W2+W3+W4，得到一个0~W-1的数i。
将i从左到右依次和第二行的数进行比较。
如果i小于该数，就对应取出那个任务。