Project4 debug 记录

马月骁 2019K8009915025

为什么只有 debug 记录而没有设计文档？自然是 oslab 没有给我剩余那么多的时间(也懒得写了)。

实际上在 P4 开发中遇到了很多 bug，这里只记录一些令人印象深刻的（有些没有即时记录，忘记了)。

以下问题中标红的为没有找到（真正）原因的。但并不意味没有办法可以绕过 bug。

问题 1：S-core（Task1)：同时开启时钟中断和切到 pid0 卡死。
原因：gdb 逐步调试发现 inst page fault 触发，进一步定位为 set_attribute 置位失败
解决方法：换种写法

```
static inline void set_attribute(PTE *entry, uint64_t bits)
{
    // TODO:
    //PTE ppn = *entry;
    //*entry = ppn;
    *entry |= bits;
}
```

原先写法即为注释中的写法。本质上两种写法没有什么区别，查看汇编也只是栈多开了 16（有一个临时变量 PPT)。至于这点区别可能为什么会导致置位失败不是很能理解，或许是触发了其他错误。进一步原因不明。

问题 2：A-core（Task2)：单核正常，双核卡死。
问题原因：初始化 shell 时有部分新添加的 pcb 信息忘记初始化了（单核没有跑其他进程前不会更换 shell 的页表)
解决方法：init_pcb 函数里面补上对应初始化代码即可。

```
        pcb[i].father = &pcb[i];
```

其实就是一行。

问题 3：A-core（Task4)：创建线程后，新建线程传参出现错误（地址变成内核的)
问题原因：线程创建时，初始化线程栈中 gp 在内核态，而线程不会进入 crt0.S，故 gp 不会更新为用户态对应值，导致传参出错。
解决方法：初始化时使用父进程的 gp

```
    init_pcb_stack(new_pcb->kernel_sp, new_pcb->user_sp,
(ptr_t)start_routine, arg, NULL, new_pcb);
    regs_context_t *spt_regs = (regs_context_t
*)(new_pcb->kernel_stack_base - sizeof(regs_context_t));
    regs_context_t *fpt_regs = (regs_context_t
*)(current_running->kernel_stack_base - sizeof(regs_context_t));
    spt_regs->regs[3] = fpt_regs->regs[3];
```

问题 4：A-core（Task4)：mailbox 行为不正常，接收、输入不匹配
问题原因：似乎是调整 make 的时候（开始写 P4 时)，为了通过编译，修改了测试文件（我

用户态的 mailbox 是一个结构体，并非 int)，导致 mq 数组值被覆盖。
解决方法：根据逻辑还原一下就好

```
mailbox_t mq[2];
for (i = 0; i < 2; ++i) {
    mq[i] = mbox_open(other_mailbox_id[i]);
}
```

只能说这是 P3 的锅。

问题 5：A-core（Task2）：lock 的系统调用仅仅只是换了一个名字，系统调用号一样也跑不起来。
原因：未知。
解决方法：套壳使用原先的（毕竟内核都没有变，系统调用是一样的）。

问题 6：A-core（Task3）：swap 在 qemu 上双核上一直失败，上板正常。
原因：image 比较小？block 对应关系不同？具体原因不明。
解决方法：不管，反正上板是对的。

问题 7：C-core：双核切 pid0 卡死
原因：没有查出来，由于时钟中断的关系，每次查错误点出现的地方都不一样，怀疑是哪个初始化没有做好，或者指针跑飞了。
解决方法：给一个没啥实际意义的用户进程，不切到 pid0 就好。

问题 8：上板时第一次 make floppy 之后连上板子总会卡死在读取内核结束。
原因：未知。
解决方法：再 floppy 一次就好。

感谢贾文庆助教帮我查出诸多问题（1，2，3 等）。
感谢宗吉祥为我指点迷津。