操作系统LAB 1实验报告

2019011215 任彦羽

功能简述

- 在TaskControlBlock结构体中增加了是否初次执行、初次执行时间和syscall调用次数三个参数，并可以调用接口/进行任务切换时对这些数值进行维护；
- 提供更新当前任务块的系统调用次数和返回当前任务TaskControlBlock的接口；
- 在每次调用 syscall 时，调用更新相应接口更新系统调用次数。

简答题

- 正确进入 U 态后，程序的特征还应有：使用 S 态特权指令，访问 S 态寄存器后会报错。 请同学们可以自行测试这些内容， 描述程序出错行为，同时注意注明你使用的 sbi 及其版本。

所使用的 sbi 版本为 `version 0.3.0-alpha.2`

bad_address的出错行为是访问了一个地址为0x0的区域，这一部分是不在内存空间里的；

bad_instruction的出错行为是在用户态模式调用`sret`；

bad_register的出错行为是`csrr a0,sstatus`，访问了特殊寄存器`sstatus`

bad_address \ bad_instruction 和 bad_register 的执行结果依次为

```
[ERROR] [kernel] PageFault in application, bad addr = 0x0, bad instruction = 0x80400410, core dumped.
[ERROR] [kernel] IllegalInstruction in application, core dumped.
[ERROR] [kernel] IllegalInstruction in application, core dumped.
```

- L40：刚进入`__restore`时， a0 代表了什么值。请指出`__restore`的两种使用情景。

a0代表的是分配完Trap上下文后的内核栈栈顶地址，参见/trap/mod.rs中trap_handler的返回值。使用情景包括在处理完trap后返回用户态时，以及在应用启动时。

- L46-L51：这几行汇编代码特殊处理了哪些寄存器？这些寄存器的的值对于进入用户态有何意义？请分别解释。【以下括号中表示根据问题说法应位于的行数，括号外为实际在trap.S中行数】

L46（L49）：处理了 sstatus 寄存器。sstatus 寄存器的 SPP 字段保存的是当前正在处理的中断发生前 CPU 处于的特权级，用于返回时切换特权级模式。这样才可以切换回

L47（L50）：处理了 sepc 寄存器，该寄存器储存了返回的地址。

L48（L51）：处理了 sscratch 寄存器，之前指向的是用户栈，恢复为内核栈栈顶。

- L53-L59：为何跳过了`x2`和`x4`

x2 为 sp，在之后进行恢复。x4 为 tp，即为线程指针，应用程序并不会改变，无需处理。

- L63：该指令之后，sp 和 sscratch 中的值分别有什么意义？

sp 指向用户栈指针，sscratch 指向内核栈指针。

- `__restore`发生状态切换在哪一条指令？为何该指令执行之后会进入用户态？

状态切换发生在 sret 指令。指令让 CPU 将当前的特权级按照 sstatus 的 SPP 字段进行设置，并跳转到 sepc 寄存器指向的指令和其他一系列操作。由于此时 SPP 为 User Mode，因此会进入用户态。

- L13：该指令之后， sp 和 sscratch中的值分别有什么意义？

sp 指向内核栈栈顶，sscratch 指向用户栈栈顶。

- 从 U 态进入 S 态是哪一条指令发生的？

发生中断异常的那条指令。

难度/工作量

难度相对较低，就是框架的bug有点小多，为了验证不是自己写的代码的问题浪费了一点时间~不过感谢助教都及时填了坑。