Report

一. 功能简介

src/task/task.rs:为 TaskControlBlock 增加新的变量：任务开始的时间和调用的次数
src/task/mode.rs:增加 update 函数，调用次数+1；增加得到当前任务的调用次数的数组函数；增加得到当前任务的开始时间的函数。
src/syscall/mode.rs:对 syscall info 调用 src/task/mode.rs 中的 update 函数进行更新，调用次数+1.
src/syscall/process.rs:在 sys_task_info 函数中，对输入的 TaskInfo，将其的 status 更新为 running；将它的 syscall_times 更新为 src/task/mode.rs 中新增函数得到的 syscall_times；将它的 time 更新为(get_time_us()-开始时间)/1000。

二. 思考题

1. 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 （运行 Rust 三个 bad 测例 (ch2b_bad_*.rs)， 注意在编译时至少需要指定 LOG=ERROR 才能观察到内核的报错信息)， 描述程序出错行为，同时注意注明你使用的 sbi 及其版本
   答 ：

```
[ERROR][kernel] PageFault in application, bad addr = 0x0, bad instruction = 0x8040008a, core dumped.
[ERROR][kernel] IllegalInstruction in application, core dumped.
[ERROR][kernel] IllegalInstruction in application, core dumped
```

   错误一：访问地址不合法
   错误二：用户态执行 s-mode 指令
   错误三：访问寄存器出错
   sbi 版本：RustSBI version 0.2.0-alpha.4、 RustSBI-QEMU Version 0.0.1
2. 深入理解 trap.S 中两个函数 __alltraps 和 __restore 的作用，并回答如下问题:
   1) L40：刚进入 __restore 时，a0 代表了什么值。请指出 __restore 的两种使用情景。
      答：a0 代表内核栈顶
      使用场景：处理完 trap 返回用户态，开始运行 app。
   2) 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。
      答：恢复 sepc,sstatus,sscratch 三个寄存器的值。
      sepc： trap 之前执行的最后一条指令的地址，即结束 trap 需要跳转的位置 。
      Sstatus：处在特权级的信息。
      sscratch：指向内核栈
   3) L53-L59：为何跳过了 x2 和 x4?
      答：
      x2：对应 sp，指向内核栈，已经保存。
      x4：对应 tp 寄存器，一般用不到。
   4) L63：该指令之后，sp 和 sscratch 中的值分别有什么意义？
      答：sp 对应的是用户栈，sscratch 是内核栈
   5) __restore：中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？

答：sret.

因为 sret 执行后，返回到异常切出去的位置并且将状态转换为用户态。

6) L13：该指令之后，sp 和 sscratch 中的值分别有什么意义？

答：sp 对应的是内核栈，sscratch 是用户栈。

7) 从 U 态进入 S 态是哪一条指令发生的？

答：进入 trap.s 之后即进入 s 态。