实现功能：

维护系统调用次数：为TCB添加task\_info\_inner，内置一个桶记录每种系统调用的次数，并在每次陷入系统调用后、进入具体系统调用函数前完成维护

维护时间：利用get\_time\_us将当前时间与第一次运行时记录的时间作差

维护状态：一定为running

问答题：

1.访问非法地址，或者以相对低特权级使用高特权级的指令等情况会触发异常,错误信息如下：

[ERROR] [kernel] PageFault in application, bad addr = 0x0, bad instruction = 0x80400408, core dumped.

[ERROR] [kernel] IllegalInstruction in application, core dumped.

[ERROR] [kernel] IllegalInstruction in application, core dumped.

2.（1）a0 寄存器负责储存函数返回值,所以 a0中是trap\_handle的返回值, 即为指向任务内核栈上的trap\_context的地址

\_\_restore两种使用场景: 首次被调度为Running时进入用户态、完成中断处理后返回用户态继续执行

（2）csrw sstatus, t0 // 记录一些中断发生前的信息，如特权级

csrw sepc, t1 // 记录异常代码发生的位置或者系统调用返回用户态时的开始位置

csrw sscratch, t2 // 保存内核栈位置

（3）x2: x2是sp寄存器, 已经保存在sscratch

x4: 不需要使用,故不保存

（4）sp 和 sscratch 分别指向内核栈和用户栈（顺序可能相反）, 执行之后它们交换指向, 此处sp指向用户栈, sscratch指向进程内核栈

（5）sret指令;指令执行后:CPU 会将当前的特权级按照 sstatus 的 SPP 字段设置为 U 或者 S ；CPU 会跳转到 sepc 寄存器指向的指令。

（6）此处sscratch指向用户栈, sp指向进程内核栈

（7）ecall