功能概述

按照实验的要求，我完成了对TaskInfo结构体信息的获取. 加入的代码部分集中在 syscall/process.rs 和 syscall/mod.rs. 我还调整了任务管理器结构体的对外可见度，方便在其他文件进行调用.

简答作业

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 三个 bad 测例 (ch2b_bad_*.rs) ）， 描述程序出错行为，同时注意注明你使用的 sbi 及其版本。
    - ch2b_bad_address: 地址非法错误.
    - ch2b_bad_instruction: 在用户态使用了 sret 指令.
    - ch2b_bad_register: 用户太访问了没有读写权限的寄存器.
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.
        - 刚进入 __restore 时，a0 代表了当前任务的内核栈顶地址。
        - __restore 的两种使用情景：
            - 当通过中断进入到内核态退出时想要恢复上下文.
            - 当通过异常进入到内核态退出时想要恢复上下文.
    2. L43-L48：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.
        - sstatus: Trap 发生之前 CPU 处在哪个特权级（S/U）等信息.
        - sepc: Trap 是一个异常的时候, Trap 发生之前执行的最后一条指令的地址.
        - sscratch: 临时的寄存器, 这里在上下文切换的过程中存储了内核态和用户态栈的数值.
    3. L50-L56：为何跳过了 x2 和 x4？
        - x2/sp: 存储了内核栈的指针, 用户态的至真此时保存在 sscratch 中, 后面会最后交换过来的.
        - x4/tp: 我们并没有使用到他到，所以跳过了.
    4. L60：该指令之后，sp 和 sscratch 中的值分别有什么意义？
        - sp和sscratch指向的用户栈和内核栈的地址交换了. 此时sscratch中保存着内核栈的栈顶地址, sp指向用户栈的栈顶地址.
    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？
        - 发生在 sret 指令, 该指令发生之后
            - sepc 寄存器中的值加载到程序计数器PC中
            - sstatus 寄存器中恢复陷入前的特权级状态
    6. L13：该指令之后，sp 和 sscratch 中的值分别有什么意义？

    - sp和sscratch指向的用户栈和内核栈的地址交换了. 此时sscratch中保存着用户栈的栈顶地址, sp指向内核栈的栈顶地址.

7. 从 U 态进入 S 态是哪一条指令发生的?

    - ecall 指令

荣誉准则

1. 在完成本次实验的过程（含此前学习的过程）中，我曾分别与 以下各位 就（与本次实验相关的）以下方面做过交流，还在代码中对应的位置以注释形式记录了具体的交流对象及内容：

   无

2. 此外，我也参考了 以下资料 ，还在代码中对应的位置以注释形式记录了具体的参考来源及内容：

    - https://rcore-os.cn/rCore-Tutorial-Book-v3/
    - https://learningos.cn/rCore-Camp-Guide-2024A/

3. 我独立完成了本次实验除以上方面之外的所有工作，包括代码与文档。 我清楚地知道，从以上方面获得的信息在一定程度上降低了实验难度，可能会影响起评分。

4. 我从未使用过他人的代码，不管是原封不动地复制，还是经过了某些等价转换。 我未曾也不会向他人（含此后各届同学）复制或公开我的实验代码，我有义务妥善保管好它们。 我提交至本实验的评测系统的代码，均无意于破坏或妨碍任何计算机系统的正常运转。 我清楚地知道，以上情况均为本课程纪律所禁止，若违反，对应的实验成绩将按“-100”分计。