

**本科生毕业设计(论文)**

# 开题报告

**rCore模块化改进的设计与实现**

**Design and implementation of modular improvement of rCore**

|  |  |
| --- | --- |
| 学 院： | 计算机学院 |
| 专 业： | 计算机科学与技术 |
| 班 级： | 07111703 |
| 学生姓名： | 石文龙 |
| 学 号： | 1120173592 |
| 指导教师： | 陆慧梅 |

|  |
| --- |
| **一、选题依据**  （简述该选题的研究意义和背景，国内外研究概况和发展趋势等）  操作系统是计算机的灵魂，目前国外操作系统品牌几乎垄断了巨大的中国市场，其中在桌面端、移动端的市占率分别超过94.75%、98.86%。根据Gartner的统计数据，2018年中国的操作系统市场容量在189亿以上，其中国外操作系统品牌几乎在中国市场处于垄断地位。截至2019年8月，在中国的桌面操作系统市场领域，微软Windows的市占率87.66%，苹果OSX的市占率为7.09%，合计为94.75%;在中国的移动操作系统市场领域，谷歌Android的市占率为75.98%，苹果iOS的市占率为22.88%，合计为98.86%。  虽然当前中国的操作系统市场依旧是微软的windows+intel占据了主导地位，但是windows的闭源架构正面临着以linux为代表的开源操作系统的挑战。中国的操作系统国产化浪潮起源与二十世纪末，目前正依托于开源操作系统的开源生态以及政策东风正在快速崛起，涌现出了中标麒麟、银河麒麟、深度Deepin、华为鸿蒙等各种国产操作系统。  所以在这个时期，我们进行操作系统的学习和研究是非常可行的。但是由于随着操作系统的功能实现的增多，其复杂程度也在增加；并且在对操作系统的学习过程中，因为各个章节是被git分支隔离开的，所以完成前一个章节的实验后，在下一个章节有关前一个章节的实现内容需要复制代码过来，同样的代码用一次就需要写一次，并且如果改了某一章节的内容，后续的所有章节相同的地方都需要重新改一边，代码的复用性非常不好。  rust的模块化编程就能很好的解决上述问题，之前做不同章节的课后实验需要切换到对应的lab分支去写代码，现在只需要封装一个crate加入workspace就可以了，之前每个实验都会有大量的代码需要重复的写在每一个章节中，模块化之后这些重复的代码就可以直接引用了，大大提高了代码的复用性。防止修改了某一模块的内容，但是后续章节没有修改造成的错误出现，减少了开发者出错的可能性。 |
| **二、研究目标和内容**  （研究目标、主要内容及关键问题等）  研究目标：在实验室工作的基础上，实现对于rcore内核模块化的改进与优化。主要完善实验室现有的模块化的rcore操作系统内核，针对现有的rcore操作系统内核模块化，提出了针对每一个章节模块的独立测试增加单元测试用例和在用户态对每一个模块进行单元测试的优化方向。  主要内容：首先需要了解rust语言的使用和RISC-V架构，并且完成用rust写操作系统内核的5个实验，理解实验室当前对内核模块化的成果，在当前成果的基础上各个章节的模块进行内核态和用户态的单元测试。 |
| **三、研究方案**  （拟采用的研究方法、技术路线、实验方案及可行性分析等）  3.1已有工作  目前实现的模块化，主要有在所有的章节中复用的代码形成了单独的package，各个章节对于这些复用的代码只需要在cargo.toml的依赖中添加上需要使用到的package就可以了，不需要再像之前一样需要将这些已经写过的代码在每一章节中都重新写一遍。  成功利用了rust语言的workspace和crate，使得每一个章节是一个预期目标不同的package，做不同的章节的实验的时候，只需要封装一个crate到对应的package中，然后在运行的时候运行指定的package就可以了，不需要像之前一样做不同章节的课后实验需要到不同的分支中写代码。  并且实现了系统调用接口的模块化，即系统调用的分发封装到一个crate中，这个crate就是syscall/src/kernel/mod.rs。使得添加系统调用的模式不是为某个 match 增加分支，而是实现一个分发库要求的 trait 并将实例传递给分发库。 3.2改进方向  首先，针对目前实验室的版本只是实现了操作系统内核模块化的基本功能的问题，增加了每个章节的模块化完成之后的单元测试。增加了单元测试之后，我们能够能够进行小而集中的测试能够在隔离的环境中一次测试一个模块或者测试模块的私有接口，能够更加方便的检测出来是代码的哪个模块甚至是哪个函数出了问题。  继续完善rcore操作系统内核的模块化，对已经存在的模块化进行自己的改进，可以在分析了已有模块接口的优缺点的基础上，只进行模块接口定义的改进；也可以对模块的实现方法进行改进，并且通过测试来分析自己的实现与已有模块的实现相比有哪些优缺点。  3.3各章节单元测试  在Rust中一个测试函数的本质就是一个函数，只是需要使用test属性进行标注或者叫做修饰，测试函数被用于验证非测试代码的功能是否与预期一致。在测试的函数体里经常会进行三个操作，即准备数据/状态，运行被测试的代码，断言结果。  在各章节的src目录下，每个文件都可以创建单元测试。标注了#[cfg(test)] 的模块就是单元测试模块，它会告诉 [Rust]只在执行cargo test时才编译和运行代码。在library项目中，添加任意数量的测试模块或者测试函数，之后进入该目录，在终端中输入cargo test运行测试。但是，我们的内核是一个no\_std应用，没有标准库，而rust的测试框架会隐饰的调用test库，而test库使依赖标准库的，所以我们的内核无法使用默认框架。  因此我们需要自定义测试框架，并且幸运的使Rust支持通过使用不稳定的自定义测试框架来代替默认的测试框架，而且该功能不需要额外的依赖库，因此在我们的no\_std内核中可以使用。它的工作原理是收集所有标注了#[test\_case]属性的函数，然后将这个测试函数的列表作为参数传递给用户指定的runner函数。因此，它实现了对测试过程的最大控制。  完成了自定义测试框架之后，我们现在在\_start函数结束之后进入了一个死循环，在每次运行完cargo test后需要我们手动瑞出QEMU，非常麻烦。幸运的是QEMU支持一种名为 isa-debug-exit的特殊设备，它提供了一种从客户系统（guest system）里退出QEMU的简单方式。我们可以通过将配置关键字package.metadata.bootimage.test-args添加到我们的Cargo.toml中来达到目的。  要在控制台上查看测试输出，我们还需要以某种方式将数据从内核发送到宿主系统，发送数据的一个简单的方式是通过串行端口，QEMU可将通过串口发送的数据重定向到宿主机的标准输出或是外部文件中。并且为了查看QEMU的串行输出，我们需要使用-serial参数将输出重定向到stdout。同时为了在panic时使用错误信息来退出QEMU，我们可以使用条件编译在测试模式下使用不同的panic处理方式。由于我们使用isa-debug-exit设备和串行端口来报告完整的测试结果，所以我们不再需要QMEU窗口了。我们可以通过向QEMU传递-display none参数来隐藏弹出的窗口。  完成上述功能后我们已经有了一个可以工作的测试框架了，我们可以为我们的VGA缓冲区实现创建一些测试。  3.4实施技术方案所需的条件 Ubuntu虚拟机  Rust 版本管理器 rustup 和 Rust 包管理器 cargo  QEMU 7.0模拟器  VSCode |
| **四、研究计划及进度安排**  12.23-12.30  了解课题研究内容，撰写开题报告  12.30-1.10  实现现有的模块化rcore内核  1.10-1.17  测试console模块的test\_log函数能否打印出设定的信息，测试print宏和println宏,并写实验文档  1.17-1.24  测试linker模块的fmt和next函数,kernel-context模块的execute，new，aligned\_size函数，multislot\_portal.rs中calculate\_size，init\_transit函数，lib.rs中empty,user,thread,x,x\_mut等读取修改函数，move\_next，execute，build\_sstatus函数，syscall模块的handle函数  1.25-2.2  测试kernel-alloc中alloc函数,kernel-vm中mapper.rs中new，ans，arrive，meet，block函数，mod.rs中new root\_ppn root map\_extern maptranslate cloneself fmt 函数，visitor.rs中new ans arrive meet block函数。  2.3-2.10  测试task-manager中id.rs进程和线程new from\_usize get\_usize函数，proc\_manage.rs中new find\_next set\_managermake\_current\_suspend make\_current\_exitedadd current get\_task wait 函数，proc\_rel.rs中new add\_child del\_child wait\_any\_child wait\_child 函数，以及线程中与进程相同的函数。  2.11-2.18  测试easy-fs中bitmap.rs中decomposition new alloc dealloc函数block\_cache.rs中new addr\_of\_offset get\_ref get\_mut readmodify sync get\_block\_cache block\_cache\_sync\_all函数，efs.rs中create open root\_inode get\_disk\_inode\_posget\_data\_block\_id alloc\_inode alloc\_data dealloc\_data函数，file.rs在new lenread\_write readable writable read write 函数，layout.rs中fmt initialize is\_valid initialize is\_dir is\_file  data\_blocks \_data\_blocks total\_blocks blocks\_num\_needed get\_block\_id increase\_size clear\_size read\_at write\_at函数和 DirEntry结构中的函数  2.19-2.28  测试signal-impl的default\_action.rs中from into函数，lib.rs中new fetch\_signal fetch\_and\_remove，from\_fork is\_handling\_signal set\_action，get\_action\_ref update\_mask handle\_signals sig\_return函数，signal\_set.rs中SignalSet结构中的函数。  3.1-3.7  测试sync中condvar.rs中new signal wait\_no\_sched，wait\_with\_mutex 函数，mutex.rs中new lock unlock 函数，semaphore.rs中new up down函数，up.rs中new enter exit函数 UPIntrFreeCell结构中new exclusive\_access exclusive\_session函数。  3.8-5.1  分析各模块接口的优缺点后修改模块的接口  5.1-6月初  完成毕业论文，参加答辩 |
| **五、创新点及预期研究成果**  本项目基于实验室的rcore操作系统内核的模块化，创新点在于增加了每一章的用户态和内核态的单元测试。  研究成果为：一篇毕业论文，一份软件成果。 |
| **六、参考文献**  [1]杨德睿 单核环境的模块化Rust语言参考实现  [2]guomeng.2021年操作系统的商业化应用 国内操作系统现状与前景分析.//2021年10月20日中研网  [3]洛佳 使用Rust编写操作系统（四）：内核测试.//2019年11月07日知乎 |
| **七、指导教师意见**  同意该开题内容  **签字：**  2023年5月17日 |
| **成绩： ，占比：0.00%** |
| **八、开题审核负责人意见**  同意  签字：  2023年5月18日 |