附录 G　实验室：xv6 项目

本章介绍了与 xv6 内核相关的项目的一些想法。该内核可从麻省理工学院获得，玩起来很有趣。完成这些项目还可以使课堂上的内容与项目更直接相关。这些项目（可能除了前面两个）通常是结对完成的，这让盯着内核代码的艰巨任务变得更加容易。

G.1　简介项目

简介项目为 xv6 添加一个简单的系统调用。可能存在许多不同的任务，包括用一个系统调用来计算已发生的系统调用次数（每次系统调用计数一次），或其他信息收集的调用。学生将学习如何实际进行系统调用。

G.2　进程和调度

学生构建比默认的轮询更复杂的调度程序。可能存在许多不同的策略，包括彩票调度程序或多级反馈队列。学生将学习调度程序的实际工作方式，以及上下文切换的方式。一个附加的小任务还要求学生弄清楚，如何在退出时让进程返回正确的错误代码，并能够通过 wait()系统调用访问该错误代码。

G.3　虚拟内存简介

基本思想是添加一个新的系统调用，给定一个虚拟地址，返回已翻译的物理地址（或报告该地址无效）。这让学生可以看到虚拟内存系统如何设置页表而无需做太多艰苦的工作。另一个可能的项目是探索如何改动 xv6，让空指针引用会产生错误。

G.4　写时复制映射

该项目为 xv6 增加轻量级 fork()的能力，名为 vfork()。这个新调用不是简单地复制映射，而是将写时复制映射设置为共享面。在引用这样的页面时，内核必须相应地创建真实副本并更新页表。

G.5 内存映射

另一个虚拟内存项目是添加某种形式的内存映射文件。可能最简单的方法，是从可执行文件中惰性加载代码页。更全面的方法，是构建 mmap()系统调用和所有必要的基础结构，以便在页故障时从磁盘换入页面。

G.6 内核线程

该项目探讨如何为 xv6 添加内核线程。clone()系统调用的操作与 fork 类似，但使用相同的地址空间。学生必须弄清楚如何实现这样的调用，以及如何创建真正的内核线程。学生还应该在其上构建一个小的线程库，提供简单的锁。

G.7 高级内核线程

学生在其内核线程之上构建一个完整的线程库，添加不同类型的锁（自旋锁，在处理器不可用时休眠的锁）以及条件变量。还要添加必需的内核支持。

G.8 基于范围的文件系统

第一个文件系统项目为基本文件系统添加了一些简单的功能。对于 EXTENT 类型的文件，学生将 inode 更改为存储范围（即指针—长度对）而不仅仅是指针。作为文件系统的相对简单的介绍。

G.9 快速文件系统

学生将基本的 xv6 文件系统转换为 Berkeley 快速文件系统（FFS）。学生构建一个新的 mkfs 工具，引入块组和新的块分配策略，并构建大文件异常。在更深层次上理解文件系统工作原理的基础知识。

G.10 日志文件系统

学生为 xv6 添加了一个基本的日志层。对于每次写入文件，日志 FS 会批量处理所有脏

块，并在磁盘日志中，为待写入的更新添加一条记录，然后再修改原来位置的块。通过引入崩溃点并展示文件系统始终恢复到一致状态，学生证明其系统的正确性。

G.11 文件系统检查器

学生为 xv6 文件系统构建一个简单的文件系统检查程序。学生将了解文件系统的一致性，以及如何检查文件系统。