深圳大学
Shenzhen University

操作系统

操作系统大作业：part I-5

谭舜泉
计算机与软件学院

大作业Part I

- 大作业Part I 由六部分组成。六部分均为阅读xv6中文文档，理解对应的xv6源代码，并回答大作业中提出的代码理解问题。
  1. （第二周）第0章：操作系统接口；
  2. （第四周）第1章：第一个进程；
  3. （第六周）附录A/B：PC硬件及引导加载器；
  4. （第八周）第2章：页表；
  5. （第十周）第4章：锁；
  6. （第十二周）第3章：陷入、中断和驱动程序。

大作业Part I

- 注意上半学期的大作业分为两部分汇总提交。
- 其中1-3提交一份实验报告，而4-6提交另一份实验报告。

实验报告

- 实验报告要求使用深大实验报告模板填写。
- 允许两至三人组成一个小组，撰写实验报告。
- 请按照实验要求，完成实验并撰写实验报告。把实验报告电子版上传到BLACKBOARD。 本实验分组完成。每个实验小组完成一份实验报告即可。但要求小组中每个成员都把实验报告的副本上传到BLCKBOARD，并注明同组成员的名字和学号。

深圳大学
Shenzhen University

第4章：锁

- 锁提供了互斥功能， 保证某个时间点只有一个CPU能持有锁。
- 如果 xv6 只能在持有特定的锁时才能使用数据结构， 那么就能保证同一时间只有一个 CPU 能使用这个数据结构。

第4章：锁

- 结构体spinlock (spinlock.h)

```
// Mutual exclusion lock.
struct spinlock {
  uint locked;       // Is the lock held?

  // For debugging:
  char *name;        // Name of lock.
  struct cpu *cpu;   // The cpu holding the lock.
  uint pcs[10];      // The call stack (an array of program counters)
                     // that locked the lock.
};
```

- 注意这里最关键的是locked成员。

第4章：锁

- （spinlock.c）acquire获得锁，release释放锁。
- 查询和设置locked值的操作必须为原子操作。
- xv6 采用了386硬件上的一条特殊指令 xchg 。xchg 指令交换了内存中的一个字和一个寄存器的值。

```
// The xchg is atomic.
// It also serializes, so that reads after acquire are not
// reordered before it.
while(xchg(&lk->locked, 1) != 0)
  ;
```

第4章：锁

- 锁是用来确保互斥访问的一种底层机制。
- 由于使用了锁机制的程序编写仍然是个巨大的挑战，所以并发和并行至今还是研究的热点。我们最好以锁为基础来构建高级的同步队列，虽然 xv6 并没有这么做。

第4章：代码理解问题

- 我们可以看到xv6总是让持有锁的线程负责释放该锁。但有一个例外。阅读“xv6中文手册”P38“代码：调度”，请回答在调度器和被调度的进程之间，如何确保ptable.lock的锁定(acquire)和释放(release)能够两两配对？

代码理解问题

- 这些问题，如果不能理解的话也不用担心。我将在两周后的课堂上（在实验报告提交前）给予解答。