OS2018 Lab1 实验报告

姓名：陈劭源　　学号：161240004

May 10, 2018

1 函数说明

1.1 系统管理函数

1.1.1 init

函数原型 `void os->init()`

说明 初始化系统管理。在本实现中，该函数没有任何作用。

1.1.2 run

函数原型 `void os->run()`

说明 操作系统的主程序。本实现中，将创建一个空闲线程，随后调用`_yield()`进行线程调度。

1.1.3 interrupt

函数原型 `_Regset *os->interrupt(_Event ev, _Regset *regs)`

说明 中断管理程序。根据不同的中断类型分别进行不同的操作。

1.2 内存管理函数

1.2.1 init

函数原型 `void pmm->init()`

说明 初始化内存管理。本实现中，将会在堆区开始初始化一个内存表，随后的内存分配和释放将会在内存表中进行登记。

1.2.2 alloc

函数原型 `void *pmm->alloc(size_t size)`

说明 在堆区中分配一块大小为`size`的内存，并对齐。

1.2.3 free

函数原型 `void pmm->free(void *ptr)`

说明 释放内存。

1.3 线程管理函数

1.3.1 init

函数原型 `void kmt->init()`

说明 初始化线程管理。本实现中，会初始化一个线程表，用于维护当前系统中的线程。

1.3.2 create

函数原型 `int kmt->create(thread_t *thread, void (*entry)(void *arg), void *arg)`

说明 创建一个线程。该线程开始运行时，会以`arg` 为参数调用`entry` 函数。如果成功，返回 1。

1.3.3 teardown

函数原型 `void kmt->teardown(thread_t *thread)`

说明 销毁一个线程。

1.3.4 schedule

函数原型 `thread_t *kmt->schedule()`

说明 调度下一个线程。

1.3.5 spin_init

函数原型 `void kmt->spin_init(spinlock_t *lk, const char *name)`

说明 初始化一个自旋锁。

1.3.6 spin_lock

函数原型 `void kmt->spin_lock(spinlock_t *lk)`

说明 获取自旋锁。不能获取已经获取的自旋锁，也不能在锁定期间发生中断。

1.3.7 spin_unlock

函数原型 `void kmt->spin_unlock(spinlock_t *lk)`

说明 释放自旋锁。不能释放还未获取的自旋锁。

1.3.8 sem_init

函数原型 `void kmt->sem_init(sem_t *lk, const char *name, int value)`

说明 初始化一个信号量。信号量的初始值不能为负。

1.3.9 sem_wait

函数原型 `void kmt->sem_wait(sem_t *lk)`

说明 执行信号量的 P 操作。

1.3.10 sem_signal

函数原型 `void kmt->sem_signal(sem_t *lk)`

说明 执行信号量的 V 操作。

2 代码结构

```
├── .git                          # git数据
│   └── ...
├── am                            # 抽象机相关文件
│   ├── am.h                      # AM API
│   ├── amdev.h                   # AM设备API
│   ├── arch.h                    # x86体系结构相关的声明
│   ├── mbr                       # x86的主引导扇区(用于加载内核)
│   └── am-x86-qemu.a             # AM API在x86上的实现
├── build                         # 编译结果目录
│   └── ...
├── framework                     # 抽象机相关文件
│   ├── kernel.h                  # 内核相关头文件
│   ├── main.c                    # 主程序
│   └── nanos.h                   # NanOS头文件
├── include                       # 头文件
│   ├── os.h                      # 操作系统头文件
│   ├── debug.h                   # 调试用头文件
│   ├── assert.h                  # C库头文件
│   ├── ctype.h
│   ├── stdio.h
│   ├── stdlib.h
│   ├── string.h
│   └── time.h
├── src                           # 源代码
│   ├── libc                      # C库的实现
│   │   └── ...
│   ├── os.c                      # 操作系统相关函数
│   ├── main.c                    # 内存管理相关函数
│   └── kmt.c                     # 多线程相关函数
├── .gitignore                    # .gitignore文件
├── git-commit.sh                 # git追踪用脚本
├── report.pdf                    # 本实验报告
└── Makefile                      # Makefile文件
```