lab2-extra
Memory Management

王子牧 刘子豪 姜春阳

lab2-extra

- 参加本次测试需要通过物理内存管理(课下测试第一个数据组)。
- 本次题目分为**基础测试、进阶测试以及课下测试**三部分
- 其中，完全通过基础测试(通过第一组测试样例)可以得到50分。完全通过进阶测试(通过第二、第三组测试样例)在基础测试的基础上额外得到50分。其中，第二组测试样例只要编译通过即可得到10分。及格(得分大于等于60分)真的不难，希望大家都勇于尝试。实际上，拿满分也不难。
- 课下测试不做额外加分，但是，如果不通过课下测试，则得分必定为0分。

Contents

实验约束

- 不可在mm/pmap.c文件中定义与以下函数同名的函数(保险起见，任意文件中都不要定义)
  - void pm_check_downround(void) # 进阶测试 课下功能测试样例
  - void test_alloc(void) # 进阶测试 新功能测试样例
  - void test_queue(void) # 课下测试点 队列相关宏
  - void pm_check_plus(void) # 课下测试点 mm/pmap.c中
    void physical_memory_manage_check(void)函数的加强版
  - 值得注意的是，我们有部分测试并未通过额外定义函数实现。
- mm/pmap.c中的page_free函数在插入page_free_list时请使用LIST_INSERT_HEAD
- 严禁修改 init/main.c 文件中的main()函数(指，不作出语义上的修改)

基础测试

目前，在我们的小操作系统中。所有物理页面可能处于的状态有三种：使用中物理页、空闲物理页、已经被申请但目前未被使用的物理页。**值得注意的是 page_alloc() 函数所做的工作仅仅是将一个物理页从空闲页面链表中取出，并没有包括使用。**请仔细思考这三种页面的判定方法，尤其是**"使用"的定义。**虽然这个定义并不难猜出，但以防万一，请大家注意一下，Page结构体的组成。

说明：

1. 当前空闲页面，状态标记为1.
2. 已经被申请，但目前未被使用的页面(即，使用次数为0)，状态标记为2.
3. 使用中的物理页，当前使用次数不为0，状态标记为3.

现在你需要实现如下函数

- 首先，你需要修改你的page_init函数，将其修改为如下形式 void page_init(int mode);并在**pmap.h**中修改对应声明。
    - **输入mode为0时**，执行该函数后，page_free_list从表头到尾的对应物理页的下标依次**递减。**
    - **输入mode不为0时**，执行该函数后，page_free_list从表头到尾的对应物理页的下标依次**递增。**
- 在**pmap.c**中实现函数 void get_page_status(int pa); 并在**pmap.h**中添加该函数的声明。
  函数的输入是一个**物理地址**，请按照格式输出该物理页的状态信息。

  ```
  printf("times:%d,page status:%d\n",var1,var2);
  ```

  其中，var1 为这是第几次调用这个函数，**从1开始计数**，即，第一次调用这个函数时，输出的var1为1。

  hint: 全局变量 或 static

  var2 是输入物理地址对应的页面的状态标记数字(1,2,3中的某一个)

  注意，使用英文字符。

进阶测试

- 题目：

- 在操作系统刚刚启动时，我们的小操作系统使用函数alloc来分配指定字节的物理内存。我们现在实现的alloc函数会从低地址申请物理内存。

- 现要求同学们改写alloc函数，使得调用alloc函数时从高地址申请物理内存。alloc的参数个数、类型、功能不变，返回值也依旧为void型指针。（注：要求被申请内存的起始地址以align对齐，若无法申请足够的物理内存，函数仍需使用panic报错，格式为panic(”out of memorty\n”);。）要求alloc函数返回的地址是申请空间的低地址,举例如申请到的空间是[start_address,start_address+size),那么函数的返回值是start_address。建议参考原版alloc使用的函数和宏

- 同时要求在alloc改写后，物理内存管理系统可以正常运行(可以自行跑跑课下代码中的void physical_memory_manage_check(void)进行测试)。

完成与提交实验的方式

1. 使用 git branch 检查并确保自己在lab2分支下
2. 将本地改动提交，避免分支切换造成混乱(add与commit操作)
3. git checkout -b **lab2-extra # 新建并切换到 lab2-extra 分支下**
4. **在lab2-extra分支下完成实验内容**
5. git add --all 或 git add -A
6. git commit -m '114154'
7. git push origin lab2-extra:lab2-extra

评测过程

1. 基础测试

```
remote: [ Test low addr to high addr list with get status ]
remote: [ PASSED:45 ]
remote: [ TOTAL:45 ]
```

2. 进阶测试：保证物理内存管理可用

```
remote: [ test whether system can manage physical memory with new alloc(). ]
remote: [ PASSED:2 ]
remote: [ TOTAL:2 ]
```

3. 进阶测试：检查向下对齐

```
remote: [ test new alloc(). ]
remote: [ PASSED:7 ]
remote: [ TOTAL:7 ]
```

4. 课下测试

```
remote: [ test under class. we check queue macro and other pm manage function required you finish under class. ]
remote: [ if you don't pass this test you will get 0 points. It's exciting! ]
remote: [ PASSED:1 ]
remote: [ TOTAL:1 ]
remote: [ You got 100 (of 100) this time. Sun Mar 29 19:46:26 CST 2020 ]
```

Good Luck!