# 实验三：UNIX V6++存储管理部分

## 1. 实验目的

结合课程所学知识，通过阅读和分析 UNIX V6++中关于内存管理部分的源代码，进一步了解操作系统内存管理模块的设计与实现细节。

## 2. 实验设备及工具

已配置好 UNIX V6++运行和调试环境的 PC 机一台。

## 3. 预备知识

（1）掌握进程执行过程中，堆栈段的变化和造成这种变化的原因。

（2）连续存储管理方式中可变分区分配的首次适应分配算法与相应的回收过程；

（3）UNIX V6++进程用户态与核心态的虚实地址构成；UNIX V6++中页表机制的构成及地址转换过程。

## 4. 实验内容

### 4.1. 代码阅读与分析

**（1）关于分页机制的数据结构**

阅读 PageDirectory.h，PageTable.h 文件，了解 UNIX V6++中关于页表项、页表、页目录项和页目录的定义。

**（2）关于进程的相对地址映射表**

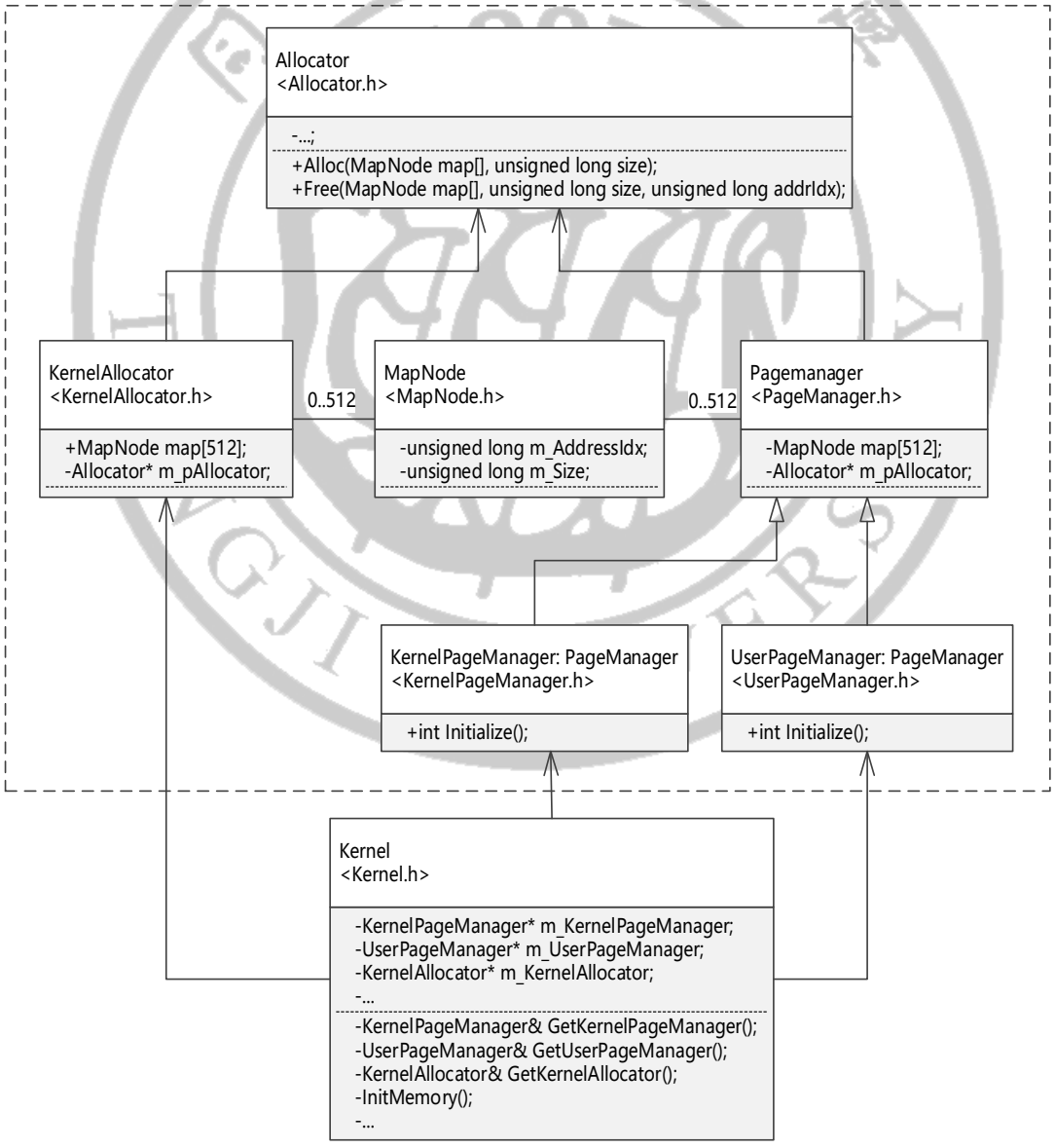
阅读 MemoryDescriptor.h 和 MemoryDescriptor.cpp，并补充完成表 1 中的黄色部分。

表 1：MemoryDescriptor 类

| 静态数据成员： | | | |
|---|---|---|---|
| 名称 | | 值 | 含义 |
| *USER_SPACE_SIZE* | | | |
| *USER_SPACE_PAGE_TABLE_CNT* | | | |
| *USER_SPACE_START_ADDRESS* | | | |
| 成员函数 | | | |
| 名称 | 输入参数 | 返回值 | 主要功能 |
| MemoryDescriptor<br>~MemoryDescriptor | 无 | 无 | 构造函数<br>析构函数 |

| Initialize | 无 | 无 | 申请并初始化 PageDirectory |
|---|---|---|---|
| Release | | | |
| MapEntry | | | |
| MapTextEntrys | | | |
| MapDataEntrys | | | |
| MapStackEntrys | | | |
| ***其他的成员函数选作*** | | | |
| 数据成员 | | | |
| 名称 | 类型 | 含义 | |
| m_UserPageTableArray | PageTable* | 指向相对虚实地址映射表 | |
| ***其他的数据成员请补充完整*** | | | |

**（3）关于空闲存储区的管理**



参照上图中各个类的继承与关联关系，分别阅读表 2 ~表 5 中的各个类或结构体相关的*.h 和*.cpp 文件，并将各表内容的黄色部分补充完整，其中函数的主要功能部分请尽量详细阐述。

表 2：MapNode 结构体

| 名称 | 类型 | 含义 |
|---|---|---|
| m_Size | | |
| m_AddressIdx | | |

表 3：Allocator 类

| 静态数据成员： | | | |
|---|---|---|---|
| 名称 | 类型 | 含义 | |
| *m_Instance* | Allocator | | |
| 静态成员函数 | | | |
| 名称 | 输入参数 | 返回值 | 主要功能 |
| GetInstance | 无 | 无 | |
| 成员函数 | | | |
| 名称 | 输入参数 | 返回值 | 主要功能 |
| Alloc | | | |
| Free | | | |

表 4：KernelAllocator 类

| 静态数据成员： | | | |
|---|---|---|---|
| 名称 | 值 | 含义 | |
| *MEMORY_MAP_ARRAY_SIZE* | | | |
| *KERNEL_HEAP_START_ADDR* | | | |
| *KERNEL_HEAP_SIZE* | | | |
| 成员函数 | | | |
| 名称 | 输入参数 | 返回值 | 主要功能 |
| KernelAllocator | Allocator* allocator | | 构造函数 |
| ~KernelAllocator | 无 | | 析构函数 |
| Initialize | | | |
| AllocMemory | | | |
| FreeMemeory | | | |
| 数据成员 | | | |
| 名称 | 类型 | 含义 | |
| map | | | |

表 5：PageManager 类及两个子类

| PageManager 类 |
|---|

| 静态数据成员： | | |
|---|---|---|
| 名称 | 值 | 含义 |
| *PHY_MEM_SIZE* | | |
| *PAGE_SIZE* | | |
| *MEMORY_MAP_ARRAY_SIZE* | | |
| *KERNEL_MEM_START_ADDR* | | |
| *KERNEL_SIZE* | | |

| 成员函数 | | | |
|---|---|---|---|
| 名称 | 输入参数 | 返回值 | 主要功能 |
| PageManager | Allocator* allocator | | 构造函数 |
| ~PageManager | 无 | | 析构函数 |
| Initialize | | | |
| AllocMemory | | | |
| FreeMemeory | | | |

| 数据成员 | | |
|---|---|---|
| 名称 | 类型 | 含义 |
| map | | |
| m_pAllocator | | |

| 子类 1：KernelPageManager | | |
|---|---|---|
| 静态数据成员： | | |
| 名称 | 值 | 含义 |
| *KERNEL_PAGE_POOL_START_ADDR* | | |
| *KERNEL_PAGE_POOL_SIZE* | | |

| 成员函数： | | | |
|---|---|---|---|
| 名称 | 输入参数 | 返回值 | 主要功能 |
| KernelPageManager | Allocator* allocator | | |
| Initialize | | | |

| 子类 2：UserPageManager | | |
|---|---|---|
| 静态数据成员： | | |
| 名称 | 值 | 含义 |
| *USER_PAGE_POOL_START_ADDR* | | |
| *USER_PAGE_POOL_SIZE* | | |

| 成员函数： | | | |
|---|---|---|---|
| 名称 | 输入参数 | 返回值 | 主要功能 |
| UserPageManager | Allocator* allocator | | |
| Initialize | | | |

UNIX V6++对空闲存储区的管理采用首次适应的连续分配方式。请仔细阅读 Allocator:: Alloc(MapNode map[], unsigned long size)函数和 Allocator:: Free(MapNode map[], unsigned long size, unsigned long addrIdx)函数，并绘制两个函数的流程图。

**（4）内存分配算法修改（选作）**

学有余力的读者可以尝试将 UNIX V6++的首次适应算法修改为循环首次适应算法，并重新编译内核。

## 5. 实验报告要求（共 4 分）

（1）（2 分）按照实验指导的要求仔细阅读相关代码，并完成表 1~表 5（所有斜体字部分为选作部分）。

（3）（2 分）绘制两个函数的流程图。

*（4）（+2 分）修改过的分配和回收函数的流程图与代码，及相应的说明。