CS302: Lab11 Report

Name：陆荻芸 SID：12011537

Answer 1

一个进程有一个`mm_struct`。`mm_struct`的作用是把一个页表的信息组合起来，包括`vma_struct`链表的首指针，对应的页表在内存里的指针，`vma_struct`链表的元素个数。

Answer 2

`vma_struct`的作用是储存一段连续的虚拟地址的描述，包括起始位置和这段虚拟地址对应的权限。同时里面还包含一个`list_entry_t`的成员指示这个结构体在链表中的位置。以及一个指向包含他的`mm_struct`的指针。

Answer 3

当进程去访问虚拟地址的时 MMU 没有找到对应的物理地址就会发生缺页中断。具体情况有：1. 页表中没有虚拟地址对应的 PTE，可能是该虚拟地址无效或虽然有效但没有分配物理内存页。2. 现有权限无法操作对应的 PTE。

Answer 4

Major page fault 的发生是因为访问的虚拟地址内容不在内存中，需要从外设载入。所以处理方式是将交换区中的页面重新载入内存。具体流程为：

- 首先分配一个物理页。如果物理页中存在空白页面，即可马上将空白页分配给该虚拟地址，将其加载进内存中。
- 如果没有空白页面的话则会发生页面置换。通过一定的算法选择被置换的页面。然后把这一页的内容写进外部交换区。再将外部交换区中的信息写入他对应的虚拟地址的 PTE 中。再把这个页面分配给进程。

代码对应 kern/mm/vmm.c 中的：

```
ptep = get_pte(mm->pgdir, addr, 1);  //(1) try to find a pte, if pte's
                                     //PT(Page Table) isn't existed, then
                                     //create a PT.
if (*ptep == 0) {
    if (pgdir_alloc_page(mm->pgdir, addr, perm) == NULL) {
        cprintf("pgdir_alloc_page in do_pgfault failed\n");
        goto failed;
    }
} else {
    if (swap_init_ok) {
        struct Page *page = NULL;
        swap_in(mm, addr, &page);  //According to the mm AND addr, try
                                   //to load the content of right disk page
                                   //    into the memory which page managed.
        page_insert(mm->pgdir, page, addr, perm);  //According to the mm,
                                                   //addr AND page, setup the
                                                   //map of phy addr <--->
                                                   //logical addr
        swap_map_swappable(mm, addr, page,
                           1);  //make the page swappable.
        page->pra_vaddr = addr;
    } else {
        cprintf("no swap_init_ok but ptep is %x, failed\n", *ptep);
        goto failed;
    }
}
```

图 1: Major page fault 处理