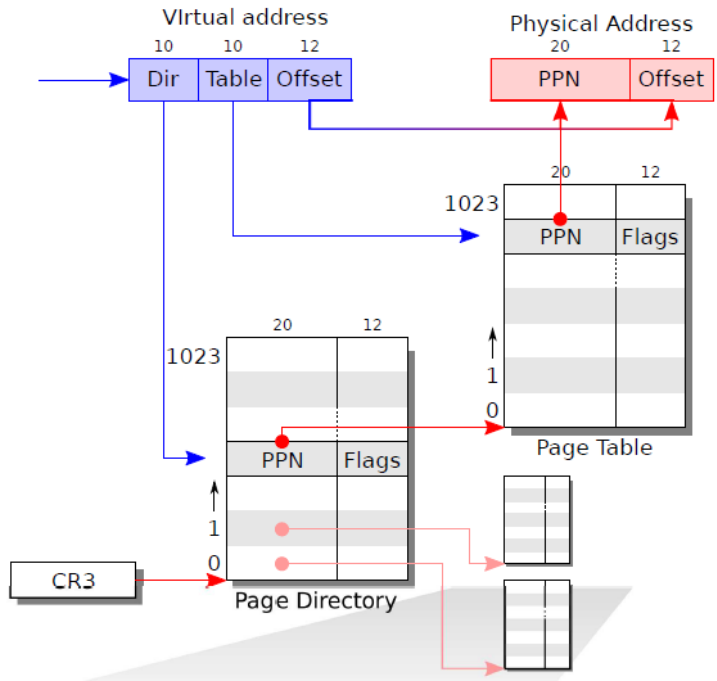
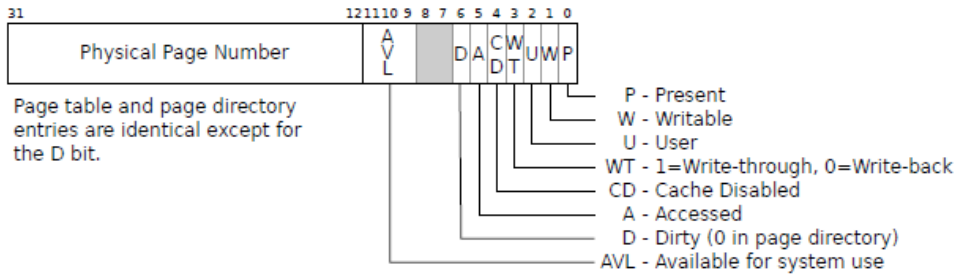
Intel 的 IA32 体系结构采用二级页表，称第一级页表为页目录(Page Directory)，第二级页表为页表(Page Table)。其虚拟地址到物理地址的翻译方式如下图。先根据 CR3 找到页目录地址，然后依据偏移 Dir 找到一个页目录项，页目录项的高 20 位(PPN)为二级页表地址；在二级页表中根据偏移 Table 找到页表项，页表项中的高 20 位(PPN)即为物理地址的高 20 位，将这 20 位与虚拟地址的低 12 位拼在一起形成完整的物理地址。



页目录和页表均有 1024 项，每一项为 4 字节，含义如下：



页目录和页表由操作系统维护，通常情况下只能在内核态下访问，为了给用户提供一个访问页表项和页目录项内容的接口，假设操作系统中已经执行过如下代码段：

```
#define UVPT 0xef400000
#define PDX(la) ((((unsigned int) (la)) >> 22) & 0x3FF)
……
kern_pgdir[PDX(UVPT)] = PADDR(kern_pgdir) | PTE_U | PTE_P;
```

其中 kern_pgdir 是操统系统维护的页目录数组，共 1024 项，每一项的类型为 unsigned int。PADDR(kern_pgdir)用于获得 kern_pgdir 的物理地址，页目录在物理内存中正好占一页，所以 kern_pgdir 的物理地址是 4KB 对齐的。PTE_U 和 PTE_P 代表了这个页目录项的权限，即用户态可访问(只读)。可以看到，这条语句将页目录的第 PDX(UVPT)项指向了页目录自身。

利用这一点，对于给定的虚拟地址 va，可以获得 va 对应的页目录项和页表项内容，分别对应于函数 get_pde 和 get_pte，请完成这两个函数（每空 2 分）：

```
#define UVPT 0xef400000
// get_pde(va)：获取虚拟地址 va 对应的一级页表(页目录)中的页目录项内容
unsigned int get_pde(unsigned int va) {
    unsigned int pdx = (va >> _[1]_____ ) & _[2]_____;
    unsigned int addr = UVPT + (_[3]_____) + pdx * 4;
                return *((unsigned int *)(addr));
}

// get_pte(va)：获取虚拟地址 va 对应的二级页表中的页表项内容
unsigned int get_pte(unsigned int va) {
    unsigned int PGNUM = va >> ___[4]_____;
    unsigned int addr = ___[5]_____ + PGNUM;
    return *((unsigned int *)(addr));
}
```