2021219113 2021213595 沈尉林

`get_physaddr` 函数解析：

1. `unsigned long pdindex = (unsigned long)virtualaddr >> 22;`
   - 这行代码将虚拟地址向右移动22位，实际上提取了虚拟地址的高10位（位31～22）。这个值用于索引页目录（PD）。
2. `unsigned long ptindex = (unsigned long)virtualaddr >> 12 & 0x03FF;`
   - 这行代码将虚拟地址向右移动12位，然后通过位与运算和0x03FF来隔离取出中间的10位（位21-12）。这个值用于索引页表（PT）。
3. `unsigned long *pd = (unsigned long *)0xFFFFF000;`
   - 这行代码将指针 `pd` 设置为固定的内存地址（0xFFFFF000），这是页目录的起始地址。这里需要检查PD条目是否存在
4. `unsigned long *pt = ((unsigned long *)0xFFC00000) + (0x400 * pdindex);`
   - 这行代码计算了页表（PT）的起始地址。它从一个固定的基地址（0xFFC00000）开始，然后根据PD索引添加一个偏移量。每个PT条目最多有 1K 个 4 字节的表项，所以偏移量是0x400（在十进制中是1024）乘以PD索引。这里需要检查PT条目是否存在。
5. `return (void *)((pt[ptindex] & ~0xFFF) + ((unsigned long)virtualaddr & 0xFFF));`
   - 这行代码执行实际的地址翻译。首先访问给定 `ptindex` 的PT条目。通过 `& ~0xFFF` 运算屏蔽这个条目的低12位（位11～0，表示页内偏移），只保留上面的位。然后，它加上虚拟地址的低12位（页内偏移）。最后返回物理地址。

下面详细解释在一个8KB页面大小的系统中，地址翻译是如何逐步进行的：

虚拟地址的结构：

1. 虚拟地址总共是32位。
2. 最高的8位（位31-24）用于索引页面目录（PD）。
3. 中间的11位（位23-13）用于索引页表（PT）。每个页表也最多可以有2048个条目，每个条目指向内存中的一个8KB页面。
4. 最低的13位（位12-0）表示页面内的偏移量。大小为8k。

翻译的步骤如下：

1. 使用虚拟地址的高8位（PD索引）查找页面目录表，获取相应的页面目录项。
2. 使用虚拟地址的中11位（PT索引）查找相应的页表，获取页表项。
3. 从页表项中获取物理页面的地址。
4. 将页面内偏移与物理页面的地址相加，得到最终的物理地址。