基础2-Linux内核空间内存申请函数kmalloc、kzalloc、vmalloc的区别

我们都知道在用户空间动态申请内存用的函数是 malloc()，这个函数在各种操作系统上的使用是一致的，对应的用户空间内存释放函数是 free()。注意：动态申请的内存使用完后必须要释放，否则会造成内存泄漏，如果内存泄漏发生在内核空间，则会造成系统崩溃。

那么，在内核空间中如何申请内存呢？一般我们会用到 kmalloc()、kzalloc()、vmalloc() 等，下面我们介绍一下这些函数的使用以及它们之间的区别。

kmalloc()

函数原型：

```
void *kmalloc(size_t size, gfp_t flags);
```

kmalloc() 申请的内存位于物理内存映射区域，而且在物理上也是连续的，它们与真实的物理地址只有一个固定的偏移，因为存在较简单的转换关系，所以对申请的内存大小有限制，不能超过128KB。

较常用的 flags（分配内存的方法）：

- **GFP_ATOMIC** —— 分配内存的过程是一个原子过程，分配内存的过程不会被（高优先级进程或中断）打断；
- **GFP_KERNEL** —— 正常分配内存；
- **GFP_DMA** —— 给 DMA 控制器分配内存，需要使用该标志（DMA要求分配虚拟地址和物理地址连续）。

flags 的参考用法：

```
|- 进程上下文，可以睡眠              GFP_KERNEL
|- 进程上下文，不可以睡眠            GFP_ATOMIC
|      |- 中断处理程序                GFP_ATOMIC
|      |- 软中断                      GFP_ATOMIC
|      |- Tasklet                   GFP_ATOMIC
|- 用于DMA的内存，可以睡眠           GFP_DMA | GFP_KERNEL
|- 用于DMA的内存，不可以睡眠         GFP_DMA |GFP_ATOMIC
```

对应的内存释放函数为：

```
void kfree(const void *objp);
```

kzalloc()

kzalloc() 函数与 kmalloc() 非常相似，参数及返回值是一样的，可以说是前者是后者的一个变种，因为 kzalloc() 实际上只是额外附加了 **__GFP_ZERO** 标志。所以它除了申请内核内存外，还会对申请到的内存内容清零。

```
/**
 * kzalloc - allocate memory. The memory is set to zero.
 * @size: how many bytes of memory are required.
 * @flags: the type of memory to allocate (see kmalloc).
 */
static inline void *kzalloc(size_t size, gfp_t flags)
{
    return kmalloc(size, flags | __GFP_ZERO);
}
```

kzalloc() 对应的内存释放函数也是 kfree()。

vmalloc()

函数原型：

```
void *vmalloc(unsigned long size);
```

vmalloc() 函数则会在虚拟内存空间给出一块连续的内存区，但这片连续的虚拟内存在物理内存中并不一定连续。由于 vmalloc() 没有保证申请到的是连续的物理内存，因此对申请的内存大小没有限制，如果需要申请较大的内存空间就需要用此函数了。

对应的内存释放函数为：

```
void vfree(const void *addr);
```

注意：vmalloc() 和 vfree() 可以睡眠，因此不能从中断上下文调用。

总结

用于申请内核空间的内存；kmalloc()、kzalloc()、vmalloc() 的共同特点是：

1. 内存以字节为单位进行分配；
2. 所分配的内存虚拟地址上连续；

kmalloc()、kzalloc()、vmalloc() 的区别是：

1. kzalloc 是强制清零的 kmalloc 操作；（以下描述不区分 kmalloc 和 kzalloc）
2. kmalloc 分配的内存大小有限制（128KB），而 vmalloc 没有限制；
3. kmalloc 可以保证分配的内存物理地址是连续的，但是 vmalloc 不能保证；
4. kmalloc 分配内存的过程可以是原子过程（使用 GFP_ATOMIC），而 vmalloc 分配内存时则可能产生阻塞；
5. kmalloc 分配内存的开销小，因此 kmalloc 比 vmalloc 要快；

一般情况下，内存只有在要被 DMA 访问的时候才需要物理上连续，但为了性能上的考虑，内核中一般使用 kmalloc()，而只有在需要获得大块内存时才使用 vmalloc()。例如，当模块被动态加载到内核当中时，就把模块装载到由 vmalloc() 分配的内存上。