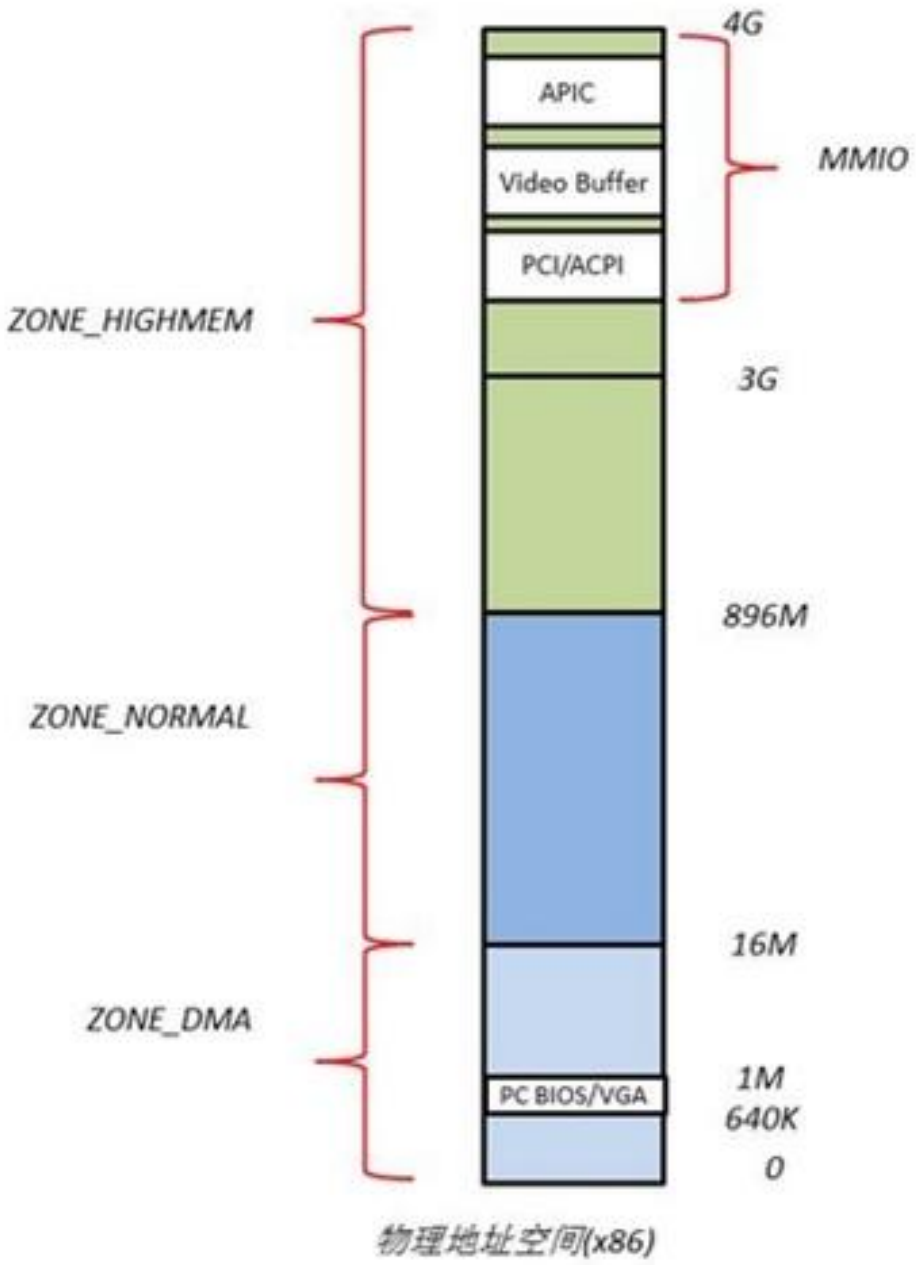
Linux 内核地址空间划分

通常 **32 位 Linux 内核**地址空间划分 0~3G 为用户空间，3~4G 为内核空间。

注意这里是 32 位内核地址空间划分，64 位内核地址空间划分是不同的。

1、x86 的物理地址空间布局：



物理地址空间的顶部以下一段空间，被 PCI 设备的 I/O 内存映射占据，它们的大小和布局由 PCI 规范所决定。640K~1M 这段地址空间被 BIOS 和 VGA 适配器所占据。

Linux 系统在初始化时，会根据实际的物理内存的大小，为每个物理页面创建一个 page 对象，所有的 page 对象构成一个 mem_map 数组。

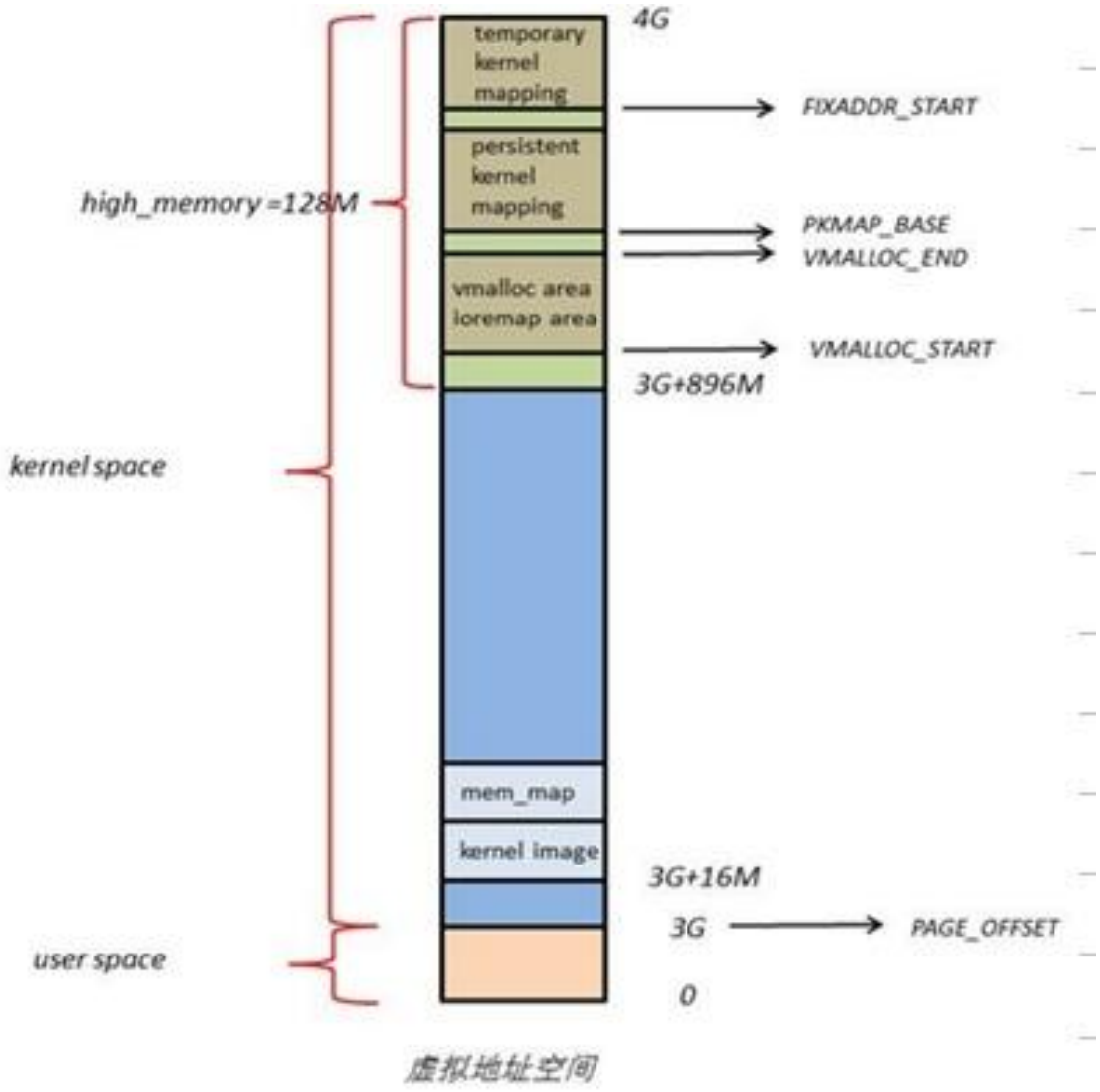
进一步，针对不同的用途，Linux 内核将所有的物理页面划分到 3 类内存管理区中，如图，分别为 ZONE_DMA，ZONE_NORMAL，ZONE_HIGHMEM。

ZONE_DMA 的范围是 0~16M，该区域的物理页面专门供 I/O 设备的 DMA 使用。之所以需要单独管理 DMA 的物理页面，是因为 DMA 使用物理地址访问内存，不经过 MMU，并且需要连续的缓冲区，所以为了能够提供物理上连续的缓冲区，必须从物理地址空间专门划分一段区域用于 DMA。

ZONE_NORMAL 的范围是 16M~896M，该区域的物理页面是内核能够直接使用的。

ZONE_HIGHMEM 的范围是 896M~结束，该区域即为高端内存，内核不能直接使用。

2、linux 虚拟地址内核空间分布

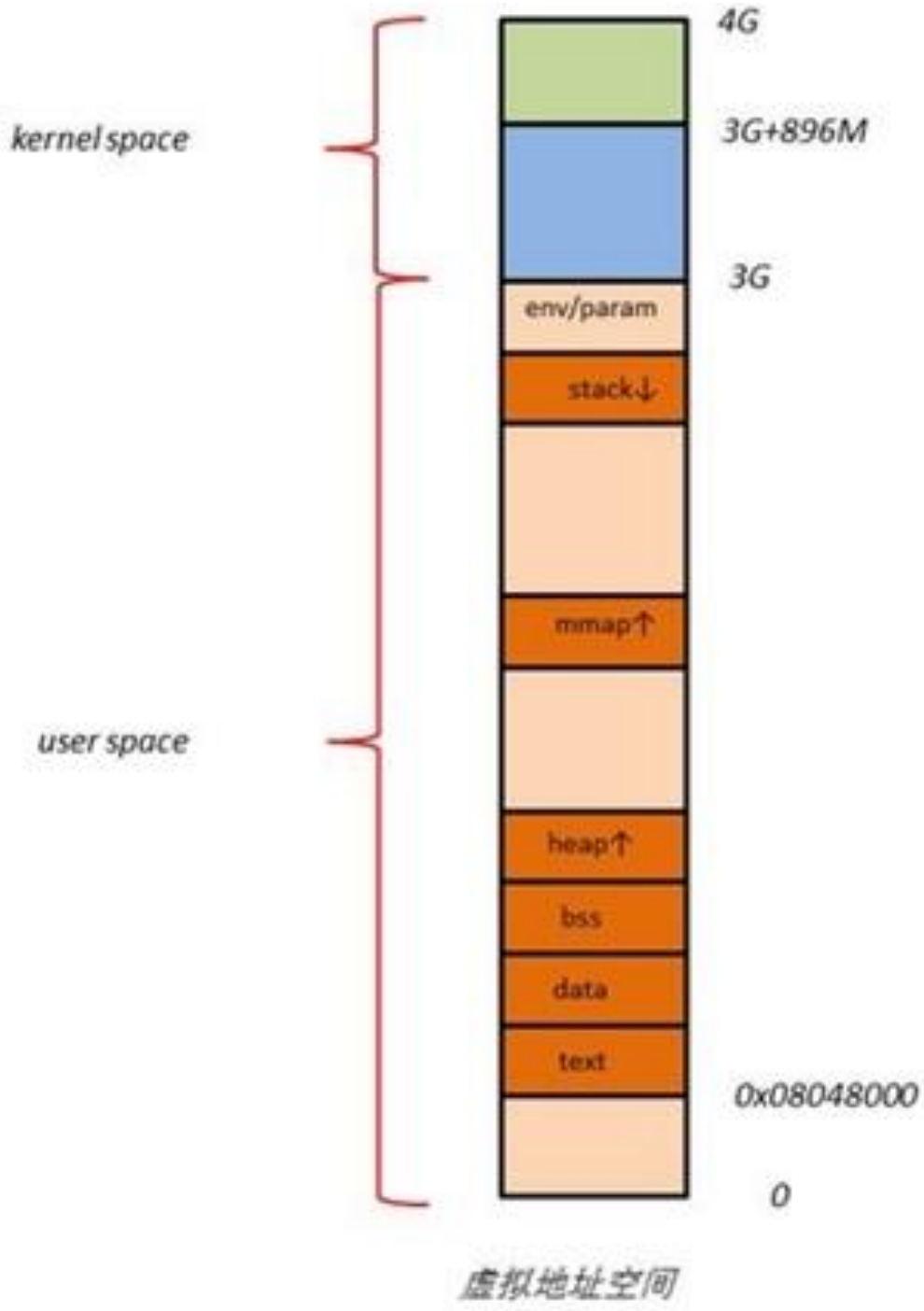


在 kernel image 下面有 16M 的内核空间用于 DMA 操作。位于内核空间高端的 128M 地址主要由 3 部分组成，分别为 vmalloc area，持久化内核映射区，临时内核映射区。

由于 ZONE_NORMAL 和内核线性空间存在直接映射关系，所以内核会将频繁使用的数据如 kernel 代码、GDT、IDT、PGD、mem_map 数组等放在

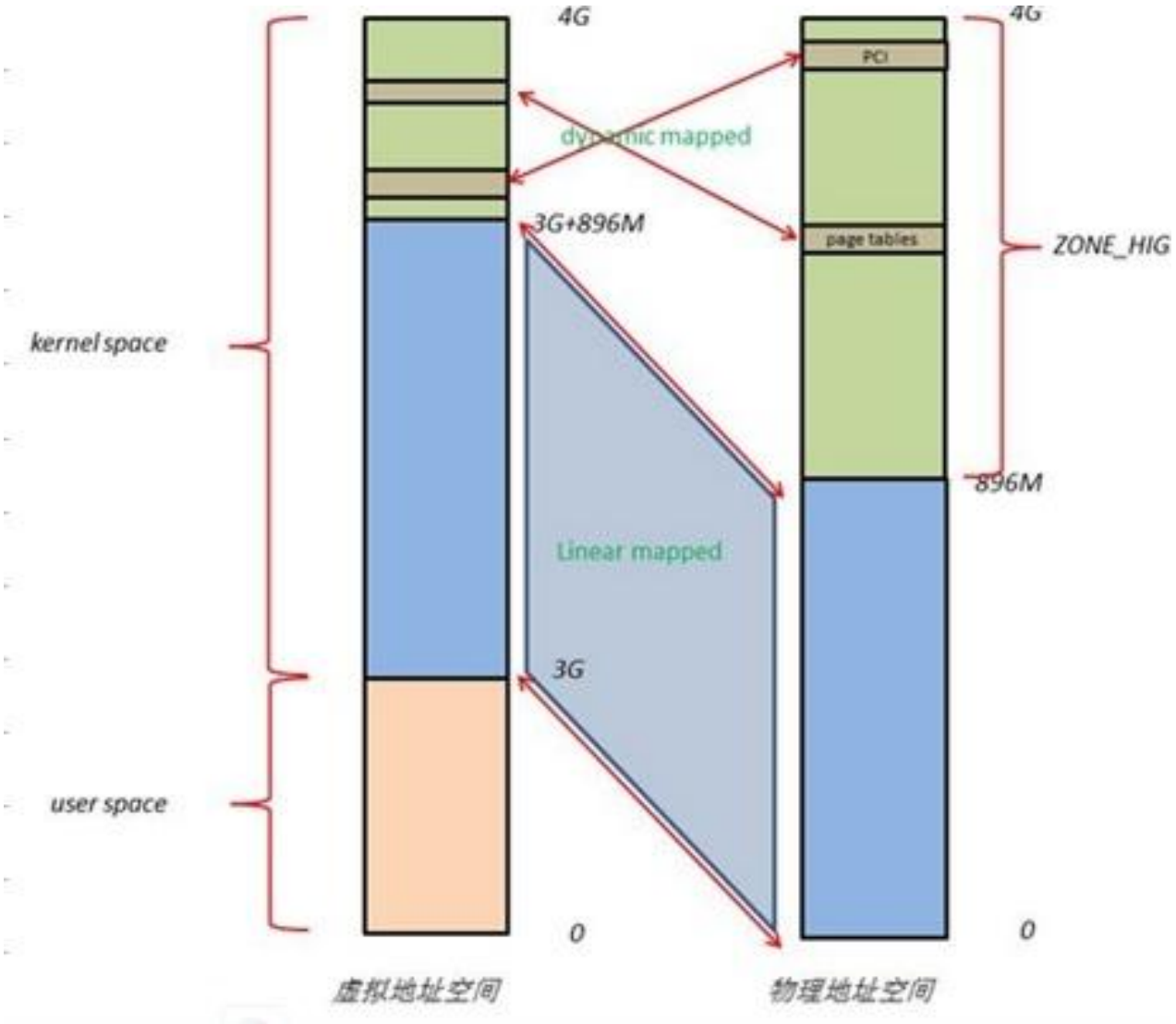
ZONE_NORMAL 里。而将用户数据、页表(PT)等不常用数据放在 ZONE_HIGHMEM 里，只在要访问这些数据时才建立映射关系(kmap())。比如，当内核要访问 I/O 设备存储空间时，就使用 ioremap()将位于物理地址高端的 mmio 区内存映射到内核空间的 vmalloc area 中，在使用完之后便断开映射关系。

3、linux 虚拟地址用户空间分布



用户进程的代码区一般从虚拟地址空间的 0x08048000 开始，这是为了便于检查空指针。代码区之上便是数据区，未初始化数据区，堆区，栈区，以及参数、全局环境变量。

4、linux 虚拟地址与物理地址映射的关系



Linux 将 4G 的线性地址空间分为 2 部分，0~3G 为 user space，3G~4G 为 kernel space。

由于开启了分页机制，内核想要访问物理地址空间的话，必须先建立映射关系，然后通过虚拟地址来访问。为了能够访问所有的物理地址空间，就要将全部物理地址空间映射到 1G 的内核线性空间中，这显然不可能。于是，内核将 0~896M 的物理地址空间一对一映射到自己的线性地址空间中，这样它便可以随时访问 ZONE_DMA 和 ZONE_NORMAL 里的物理页面；此时内核剩下的

128M 线性地址空间不足以完全映射所有的 ZONE_HIGHMEM，Linux 采取了动态映射的方法，即按需的将 ZONE_HIGHMEM 里的物理页面映射到 kernel space 的最后 128M 线性地址空间里，使用完之后释放映射关系，以供其它物理页面映射。虽然这样存在效率的问题，但是内核毕竟可以正常的访问所有的物理地址空间了。

5、buddyinfo 的理解

cat /proc/buddyinfo 显示如下：

Node 0, zone DMA 0 4 5 4 4 3 ...

Node 0, zone Normal 1 0 0 1 101 8 ...

Node 0, zone HighMem 2 0 0 1 1 0 ...

其中，Node 表示在 NUMA 环境下的节点号，这里只有一个节点 0；zone 表示每一个节点下的区域，一般有 DMA、Normal 和 HignMem 三个区域；后面的列表示，伙伴系统中每一个 order 对应的空闲页面块。