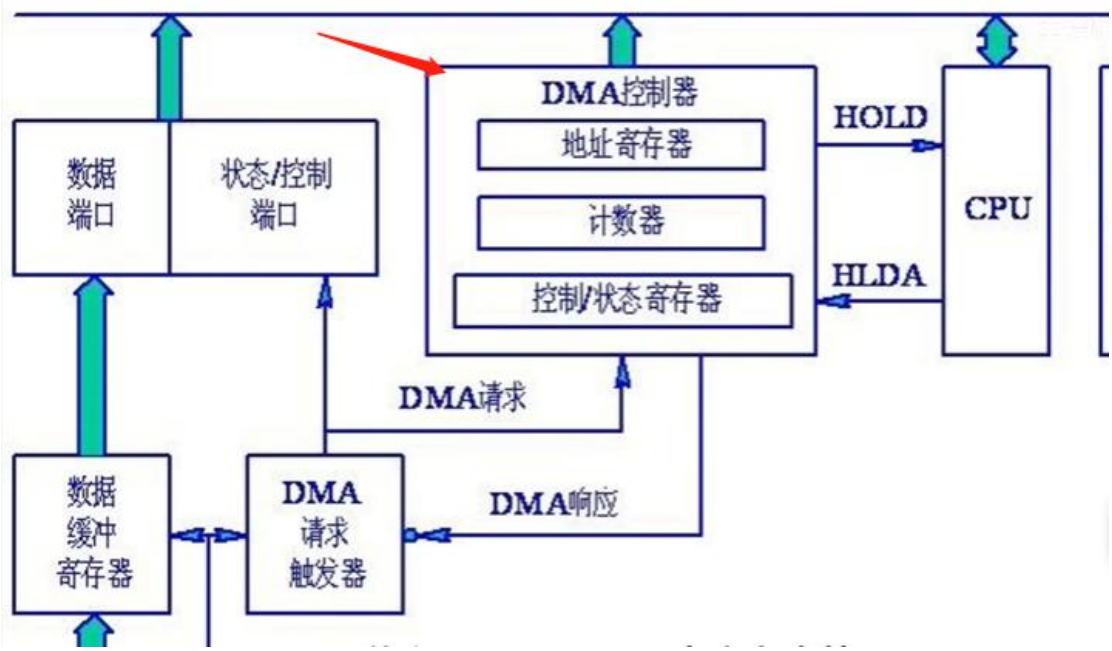
零拷贝原理

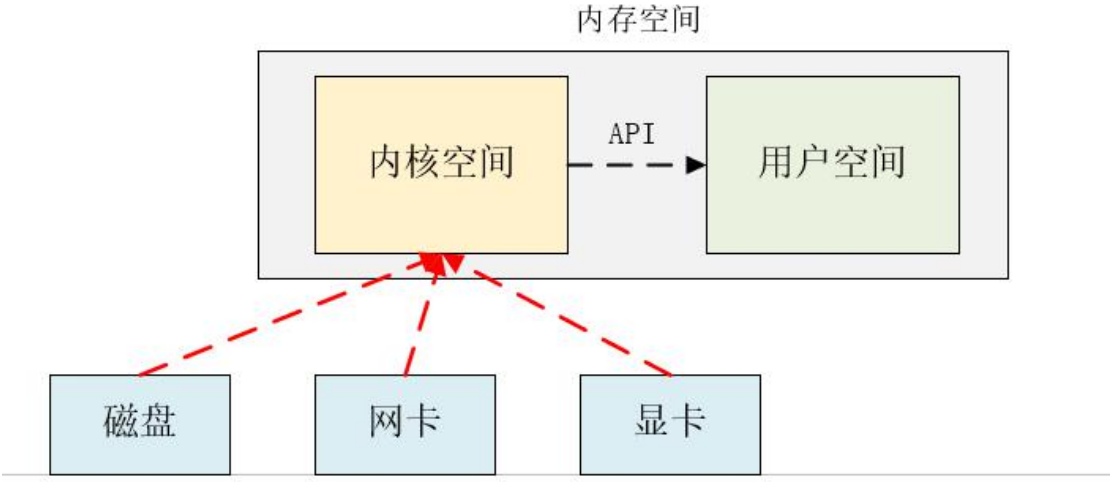
一、DMA 介绍

DMA 又叫直接内存访问（Direct Memory Access），是独立于 CPU 的一个电子元器件，可以在不消耗 CPU 性能的情况下，进行内存和外用设备（磁盘，显示器，网卡等）数据的沟通



二、数据的内核态和用户态

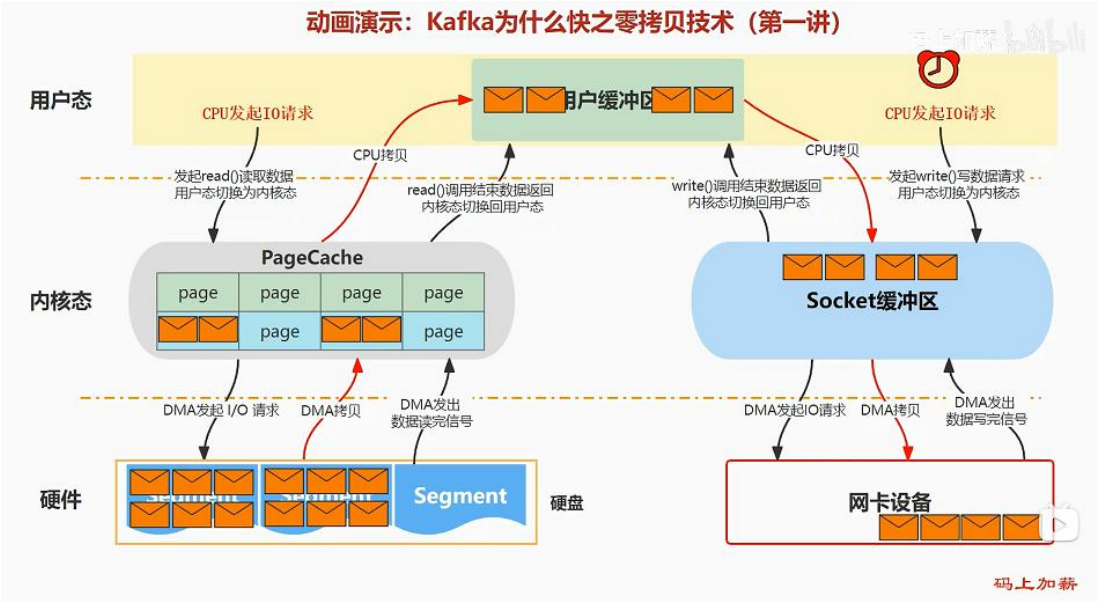
1. 为防止用户代码对计算机系统造成严重的损坏，计算机系统不允许用户代码直接调用系统的磁盘文件、网卡，显卡等，而是通过计算机系统给定的 API 来进行间接调用。
2. 这就导致计算机内存（RAM）分为大致的两块：用来存储用户代码的用户空间（如 JVM）其中数据在内核空间称为内核态，在用户空间称为用户态



下面将以 Kafka 发送数据为例，讲解零拷贝的几种方式

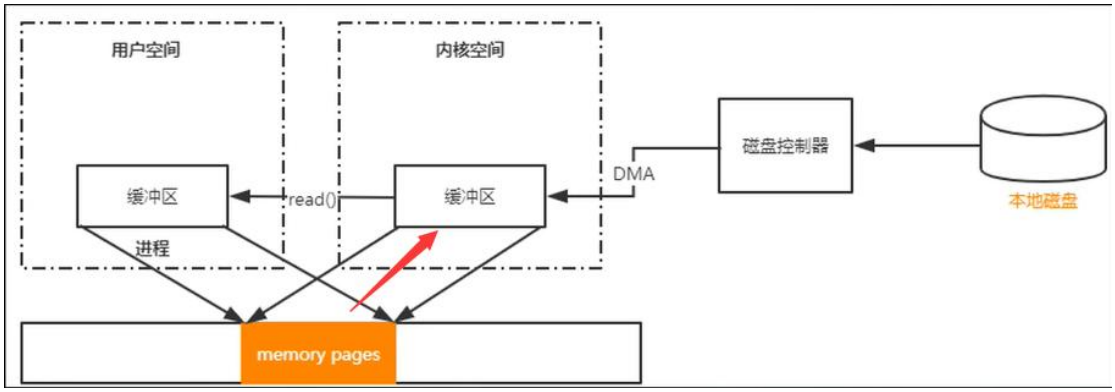
三、传统的数据拷贝过程

1. 当用户空间向内核空间发送 read（）命令时，用户线程会发生阻塞，CPU 会收到命令，并向 DMA 发送数据拷贝的命令；
2. DMA 收到数据拷贝命令后，会将外接设备（磁盘，网卡等）中的数据，拷贝至内核空间中，并通知 CPU 数据已经拷贝完成；
3. 然后 CPU 会将内核空间的数据，拷贝至用户空间，并通知用户空间，数据已拷贝完毕；
4. 用户空间收到拷贝完毕的指令后，线程由阻塞状态变为运行状态，并由用户空间向内核空间发送 write()指令；
5. 内核空间在收到 write()指令后，CPU 会将用户空间的数据拷贝至内核空间的 Socket 缓冲区，并发送指令给 DMA；
6. DMA 将内核空间 Socket 缓冲区的数据，拷贝至外接设备（如磁盘、网卡等）
7. 整个过程发生了 4 次数据拷贝，2 次 CPU 拷贝，2 次 DMA 拷贝，效率较低



四、零拷贝方式---mmap 拷贝

1. MMAP 其实就是将内核空间的数据，同用户空间的虚拟内存地址进行映射，以达到内核空间和用户空间共享内存数据的效果，来减少拷贝次数的目的。



2. 利用 mmap 函数进行零拷贝的过程如下：

① 首先，用户空间会向内核空间发送 mmap 函数，同时 mmap 函数会将用户空间的虚拟内存和内核空间的内存建立映射关系；

② 同时，CPU 在收到 mmap 指令后，会向 DMA 发送指令，将外接设备（磁盘，网卡）

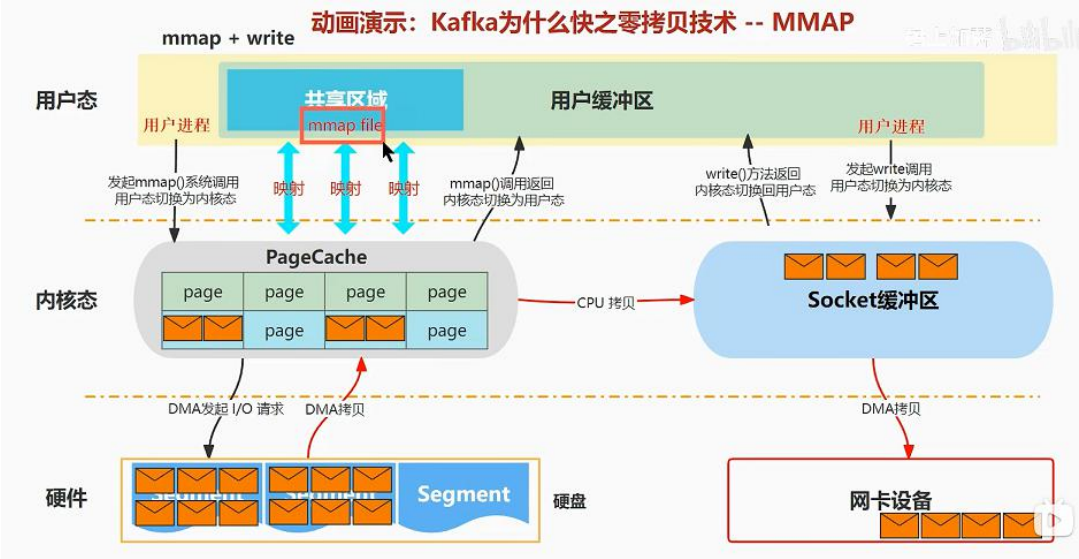
等的数据拷贝至内核空间中；

③ 然后 DMA 向 CPU 发送拷贝完毕相应，CPU 向用户空间发送拷贝完毕指令；

④ 然后用户空间向内核空间发送 write 指令，CPU 在接收到 write 指令后，会直接在内核空间中，将数据拷贝至 socket 缓冲区中；

⑤ 然后 CPU 向 DMA 发送指令，由 DMA 将数据拷贝至网卡等设备中

⑥ 整个过程只有 3 次拷贝，1 次 CPU 拷贝，2 次 DMA 拷贝,4 次上下文切换（用户态，内核态之间切换）



五、零拷贝方式--sendfile

1. sendfile 是在 mmap 的基础上，减少了上下文切换的次数（即内核态和用户态之间的切换），来提高效率。（上下文切换也是消耗时间的）
2. 具体过程如下：

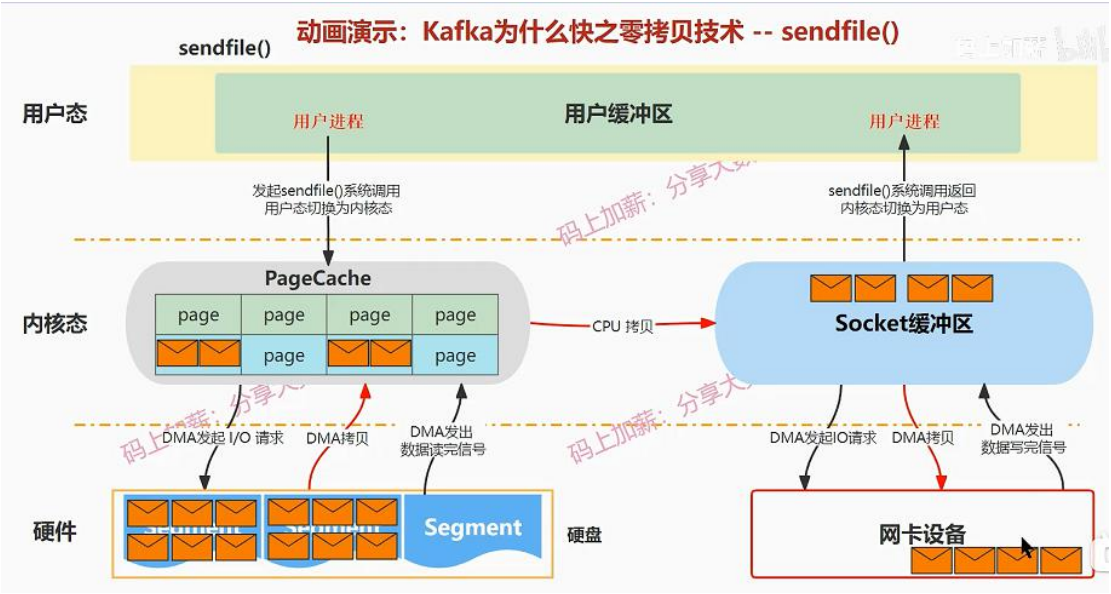
① 用户空间向内核空间发送 sendFile 函数；

② CPU 在收到 sendFile 函数后，向 DMA 发送指令，由 DMA 将数据从外接设备（磁盘，网卡等）拷贝至内核空间；

③ 然后 CPU 会直接将内核空间的数据拷贝至 socket 缓冲区，并向 DMA 发送指令，由 DMA 将数据发送至外接设备（磁盘，网卡等）；

④ 然后 DMA 返回拷贝完成响应，CPU 向用户空间返回 sendFile 执行完成相应；

⑤ 整个过程 3 次数据拷贝，1 次 CPU 拷贝，2 次 DMA 拷贝，2 次上下文切换



六、零拷贝方式---sendFile+DMA gather 拷贝

1. 在 Linux2.4 之后，对 sendFile 函数进行了改进，数据拷贝过程无 CPU 参与，具体过程如下：

① 用户空间向内核空间发送 sendFile 函数；

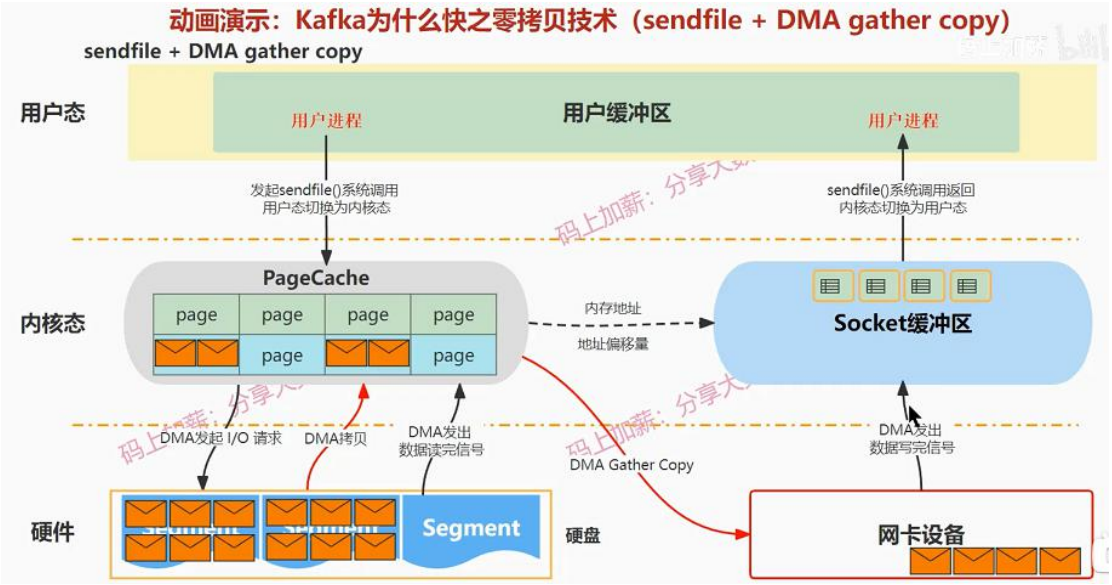
⑥ CPU 在收到 sendFile 函数后，向 DMA 发送指令，由 DMA 将数据从外接设备（磁盘，网卡等）拷贝至内核空间；

② 然后 DMA gather 技术会将内核空间数据的地址以及偏移量，发送到 socket 缓冲区

③ 然后 DMA 就会根据 socket 缓冲区中的数据地址以及偏移量，将内核空间中的数据拷贝至外接设备（网卡，磁盘等）

④ 然后 DMA 返回拷贝完成响应，CPU 向用户空间返回 sendFile 执行完成相应；

⑤ 整个过程 2 次 DMA 数据拷贝，2 次上下文切换,全程无 CPU 拷贝的参与，大大提高了效率；



七、零拷贝方式---splice

1. splice 函数就是在内核缓冲区建立数据和 socket 缓冲区的通道，由 DMA 完成数据到 socket 缓冲区的传送

2. 具体过程如下：

① 用户空间向内核空间发送 splice 函数;

② CPU 在收到 splice 函数后，向 DMA 发送指令，由 DMA 将数据从外接设备（磁盘，网卡等）拷贝至内核空间；

③ 然后 splice 函数会创建从内核空间数据到 socket 缓冲区的通道，并且由 DMA 将数据拷贝至 socket 缓冲区中，进而拷贝至网卡；

④ 然后 DMA 返回拷贝完成响应，CPU 向用户空间返回 sendFile 执行完成相应；

⑤ 整个过程 2 次 DMA 数据拷贝，2 次上下文切换,全程无 CPU 拷贝的参与