操作系统

1. 进程间通信方式

- 管道(pipe)：管道是一种半双工的通信方式，数据只能单向流动，而且只能在具有亲缘关系的进程间使用。进程的亲缘关系通常是指父子进程关系。
- 有名管道 (named pipe)： 有名管道也是半双工的通信方式，但是它允许无亲缘关系进程间的通信。
- 信号量(semophore)： 信号量是一个计数器，可以用来控制多个进程对共享资源的访问。不是用于交换大批数据,而用于多线程之间的同步.常作为一种锁机制,防止某进程在访问资源时其它进程也访问该资源。因此，主要作为进程间以及同一进程内不同线程之间的同步手段。
- 消息队列(message queue)： 消息队列是由消息的链表，存放在内核中并由消息队列标识符标识。消息队列克服了信号传递信息少、管道只能承载无格式字节流以及缓冲区大小受限等缺点。
- 信号 (signal)： 信号是一种比较复杂的通信方式，用于通知接收进程某个事件已经发生。
- 共享内存(shared memory)： 共享内存就是映射一段能被其他进程所访问的内存，这段共享内存由一个进程创建，但多个进程都可以访问。共享内存是最快的 IPC 方式，它是针对其他进程间通信方式运行效率低而专门设计的。它往往与其他通信机制，如信号两，配合使用，来实现进程间的同步和通信。
- 套接字(socket)： 套解口也是一种进程间通信机制，与其他通信机制不同的是，它可用于不同机器间的进程通信。

2. 操作系统的特征

- 并发：并发是指两个或多个事件在同一时间间隔内发生。操作系统的并发性是指计算机系统中同时存在多个

运行着的程序，因此它应该具有处理和调度多个程序同时执行的能力。

- 共享：共享是指系统中的资源（硬件资源和信息资源）可以被多个并发执行的程序共同使用，而不是被其中一个独占。资源共享有两种方式：互斥访问和同时访问。
- 异步：在多道程序环境下，允许多个程序并发执行，但由于资源有限，进程的执行不是一贯到底，而是走走停停，以不可预知的速度向前推进，这就是进程的异步性。
- 虚拟：虚拟是一种管理技术，把物理上的一个实体变成逻辑上的多个对应物，或把物理上的多个实体变成逻辑上的一个对应物的技术。采用虚拟技术的目的是为用户提供易于使用、方便高效的操作环境。

3. 进程的组成

3.1. 进程控制块（Process Control Block，PCB）

- 进程描述信息：进程标识符（PID）、用户标识符（UID）。
- 进程控制和管理信息：进程当前状态、进程优先级、代码运行入口地址、程序的外存地址、进入内存时间、处理器占用时间、信号量使用。
- 资源分配清单：代码段指针、数据段指针、堆栈段指针、文件描述符、键盘、鼠标。
- 处理器相关信息：通用寄存器值、地址寄存器值、控制寄存器值、标志寄存器值、状态字

3.2. 程序段

3.3. 数据段

4. 线程与进程的比较

- 调度：在传统的操作系统中，拥有资源和独立调度的基本单位都是进程。在引入线程的操作系统中，线程是独立调度的基本单位，进程是资源拥有的基本单位。在同一进程中，线程的切换不会引起进程切换。在不同进程中进行线程切换,如从一个进程内的线程切换到另一个进程中的线程时，会引起进程切换。

- 拥有资源：不论是传统操作系统还是设有线程的操作系统，进程都是拥有资源的基本单位，而线程不拥有系统资源（也有一点必不可少的资源），但线程可以访问其隶属进程的系统资源。

- 并发性：在引入线程的操作系统中，不仅进程之间可以并发执行，而且多个线程之间也可以并发执行，从而使操作系统具有更好的并发性，提高了系统的吞吐量。

- 系统开销：由于创建或撤销进程时，系统都要为之分配或回收资源，如内存空间、 I/O设备等，因此操作系统所付出的开销远大于创建或撤销线程时的开销。类似地，在进行进程切换时，涉及当前执行进程CPU环境的保存及新调度到进程CPU环境的设置，而线程切换时只需保存和设置少量寄存器内容，开销很小。此外，由于同一进程内的多个线程共享进程的地址空间，因此，这些线程之间的同步与通信非常容易实现，甚至无需操作系统的干预。

- 地址空间和其他资源（如打开的文件）：进程的地址空间之间互相独立，同一进程的各线程间共享进程的资源，某进程内的线程对于其他进程不可见。

- 通信方面：进程间通信(IPC)需要进程同步和互斥手段的辅助，以保证数据的一致性，而线程间可以直接读/写进程数据段（如全局变量）来进行通信。

5. 死锁产生的必要条件

产生死锁必须同时满足以下四个条件，只要其中任一条件不成立，死锁就不会发生。

- 互斥条件：进程要求对所分配的资源进行排他性控制，即在一段时间内某资源仅为一个进程所占有。此时若有其他进程请求该资源，则请求进程只能等待。
- 不剥夺条件：进程所获得的资源在未使用完毕之前，不能被其他进程强行夺走，即只能由获得该资源的进程自己来释放。
- 请求和保持条件：进程每次申请它所需要的一部分资源，在等待新资源的同时，进程继续占有已分配的资源。
- 循环等待条件：存在一种进程资源的循环等待链，链中每一个进程已获得的资源同时被链中下一个进程所请求。

6. Linux文件系统

一个典型的Linux分区如下图所示：

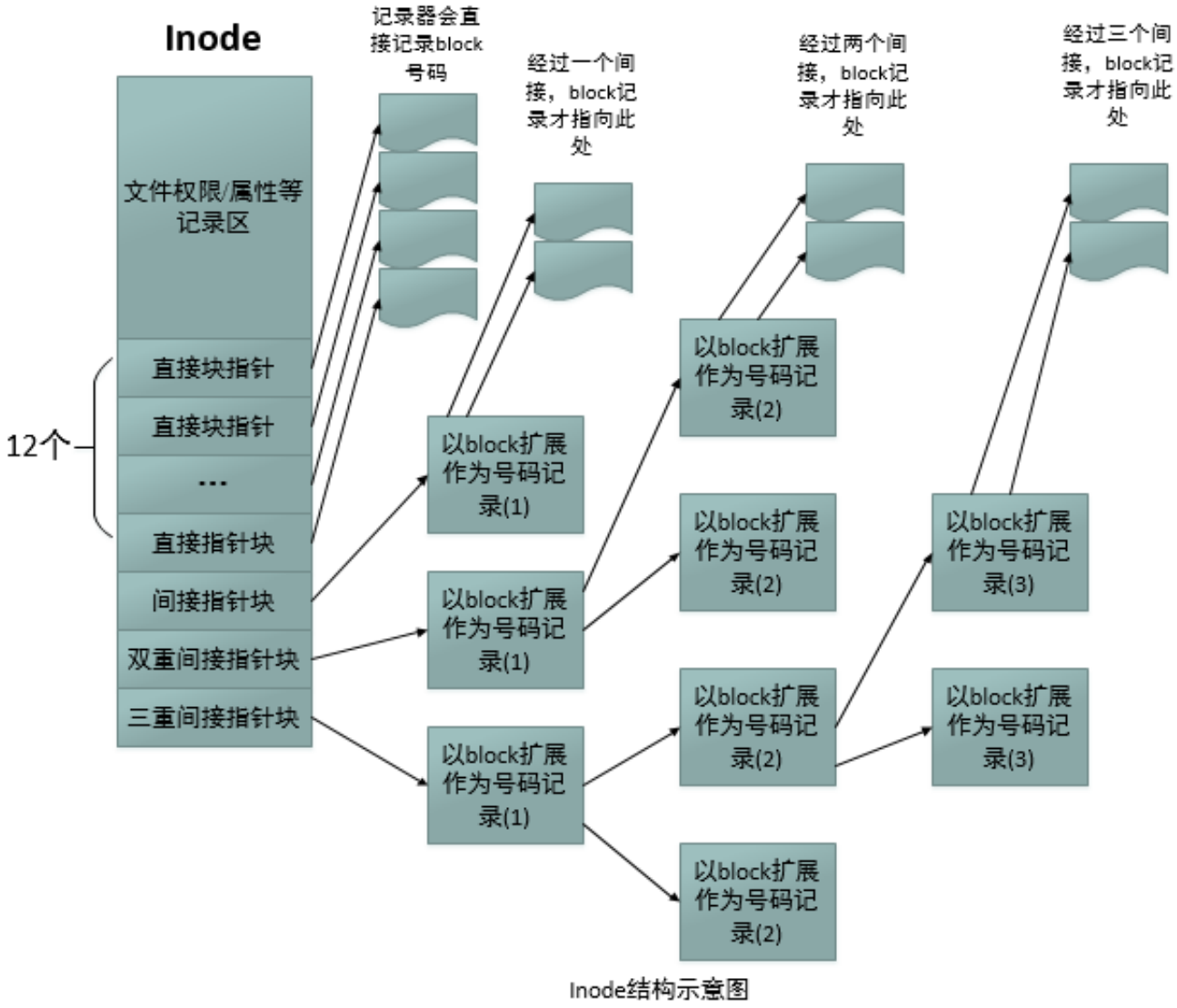
分区的第一个部分是启动区（Boot Block），主要是为计算机开机服务的。

启动区之后是超级区（Super Block），记录了文件系统的相关信息，如：

- block与inode的总量
- block与inode的大小
- 文件系统的挂载时间、最近一次写入数据的时间、最近一次检验磁盘的时间等文件系统的相关信息
- 一个validbit数值，若此文件系统已被挂载，则validbit为0，否则为1

超级区之后是多个inode，每个inode的大小均固定为128B或256B，inode记录的文件数据有：

- 该文件的访问模式（read / write / excute）
- 该文件的所有者与组（owner / group）
- 该文件的大小
- 指向此文件内容的硬连接数
- 该文件创建或状态改变的时间（ctime）
- 最近一次的读取时间（atime）
- 最近修改的时间（mtime）



Inode结构示意图

inode之后是数据块（Data Block），其特点是：

- block的大小（1KB、2KB、4KB）与数量在格式化完就不能再改变（除非重新格式化）
- 每个block内最多只能放置一个文件的数据

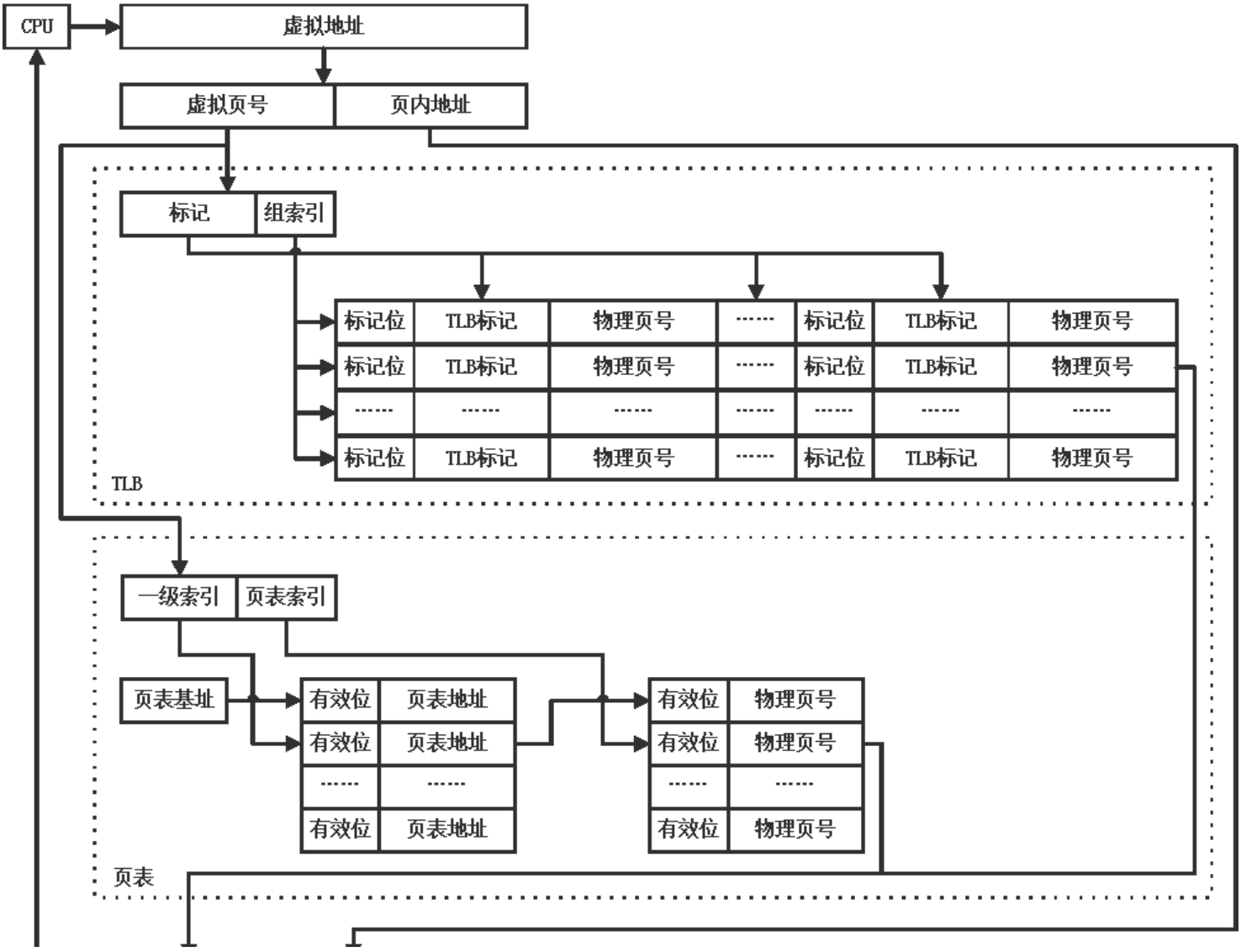
- 如果文件大于block的大小，则一个文件会占用多个block数量

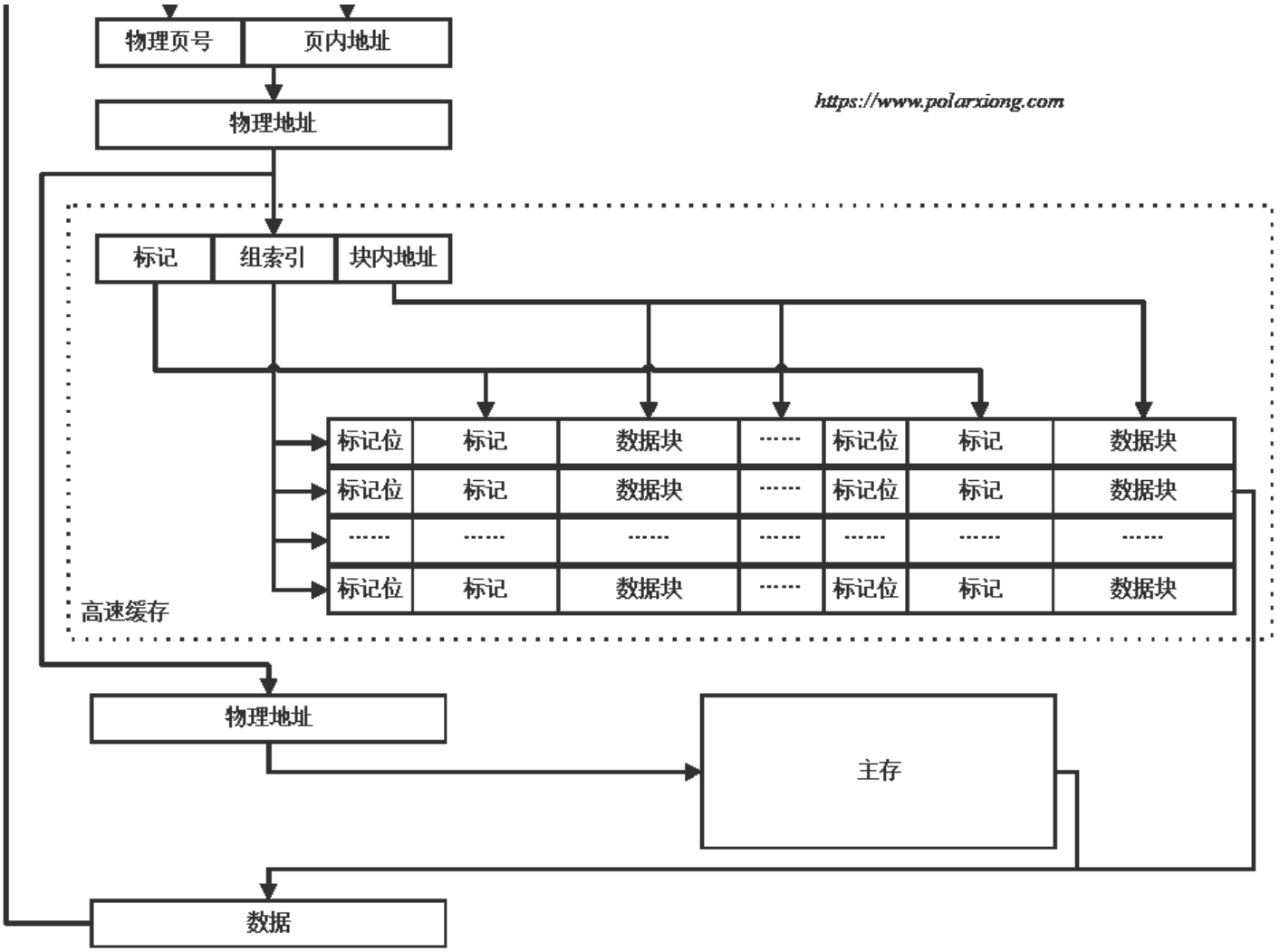
7. Linux中的文件描述符

文件描述符（file descriptor）是内核为了高效管理已被打开的文件所创建的索引，其是一个非负整数（通常是小整数），用于指代被打开的文件，所有执行I/O操作的系统调用都通过文件描述符。程序刚刚启动的时候，0是标准输入，1是标准输出，2是标准错误。

每一个文件描述符会与一个打开文件相对应，同时，不同的文件描述符也会指向同一个文件。相同的文件可以被不同的进程打开也可以在同一个进程中被多次打开。系统为每一个进程维护了一个文件描述符表，该表的值都是从0开始的，所以在不同的进程中你会看到相同的文件描述符，这种情况下相同文件描述符有可能指向同一个文件，也有可能指向不同的文件。

8. 虚拟地址到物理地址的翻译过程




物理页号
页内地址
物理地址
https://www.polarxiong.com
标记
组索引
块内地址
标记位
标记
数据块
.....
高速缓存
物理地址
主存
数据