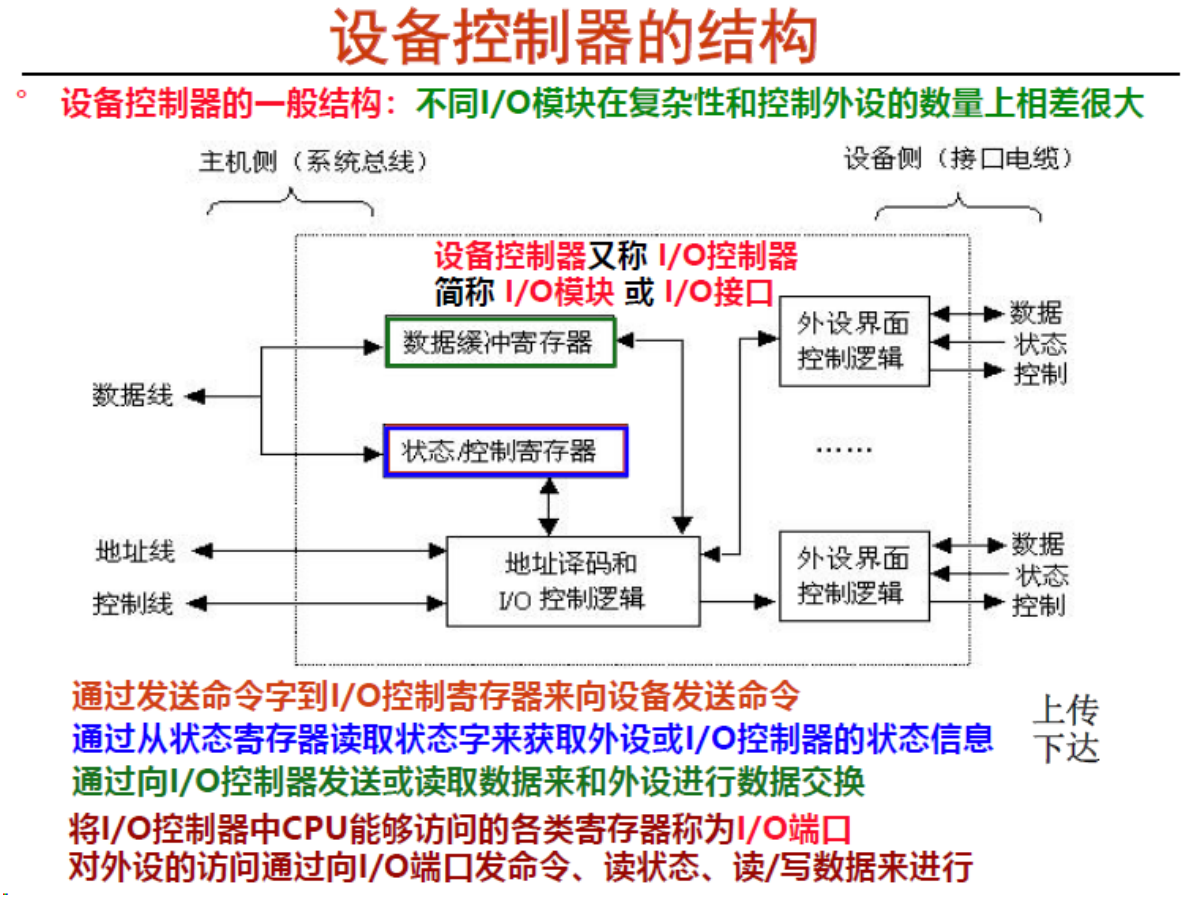
设备管理

为什么要求设备管理？I/O设备繁多，输入输出数据速度、粒度、性能各有差异，带来两个问题：1、高速CPU与低速外设所带来的CPU等待，使用率降低 2、用户使用的接口问题，CPU与外设，外设之间的并行性

为了解决这些问题，需要操作系统进行设备管理。

设备=用于操作的机械部件+用于控制的电子部件（设备控制器），CPU不会直接控制机械部件，而是通过设备控制器来控制机械部件，设备控制器为CPU提供了良好的接口。



有了接口之后便可以对其进行控制，主要是通过对设备控制器中的寄存器（控制状态寄存器、数据缓冲区）来进行读写实现的，如何读写？两种（1、独立I/O指令如IN/OUT，这是独立编址的方式； 2、内存映射I/O，也就是统一编址，这样的话就可以相当于通过对内存的读写来完成对外设的控制）

CPU与外设的数据交换有三种实现方式：A、轮询 B、中断 C、DMA

A、**对于CPU**：需要传输数据时向IO设备发出start指令，接下来便进入循环查询状态寄存器，当其状态为done时，说明数据准备完毕，可以传输数据了。**对于设备**，收到start指令后便开始准备数据，准备完毕后发出done信号，接下来传输数据。优点：实现简单，不需要额外硬件；缺点：1、CPU发出start下、指令后便进入等待状态，使得CPU利用率低 2、CPU与外设是串行的，外设之间也是串行 3、不断进行查询操作，带来较大的系统开销

B、需要另外的硬件作为支撑：设备控制器与CPU之间的中断请求线、设备控制器中的中断允许位。**对于CPU**：需要传输数据时，发出start指令后，允许中断允许位，然后该进程便进入等待状态，这时候CPU额可以选择其他进程占据，并开始执行。直到设备控制器发出中断请求信号，这时原本处于等待的进程会占据CPU，进行接下来的数据传输。**对于设备**，收到start指令后便开始准备数据，准备完毕后发出中断请求，接下来传输数据。优点：1、CPU利用率得到提高 2、并行性得以实现； 缺点：1、数据缓冲区通常较小，填满后便会中断，因此可能会频繁触发中断，带来大量系统开销 2、如果外设的速度也很快，可能会导致数据缓冲区中的数据来不及读出就丢失。

C、DMA（direct memory access？）直接内存访问。也就是直接控制外设与内存进行数据交换，不需要经过CPU。具体过程类似于中断，不过另外需要硬件：计数器+地址寄存器。DMA控制器使用总线的优先级高于CPU，通过周期窃取来完成数据的传送。**对于CPU**：发出start指令，设置计数器、地址寄存器、中断允许位，然后当前进程进入等待，其他进程占据CPU；**对于设备**，根据地址寄存器和计数器，读取数据块，全部读取完毕后通过中断请求线发出请求，唤醒目标进程。

中断是数据缓冲寄存器满之后触发中断；DMA是数据块传送完毕后触发中断。

通道技术类似于DMA，但是比DMA高级一点。DMA控制器的计数器、地址寄存器等内容的初始化需要CPU来完成，并且DMA控制器控制一个外设与内存传送数据；对于通道技术，这些工作交给通道来完成，并且一个通道可以管理多个外设与内存的传送数据。

设备的独立性，也就是在编写程序的时候要做到：程序与具体哪一台设备无关；程序尽可能与哪一类设备无关。

为了实现设备的独立性，可以通过：1、虚拟设备（虚拟设备与实际设备的映射） 2、统一命名，统一编址，也就是将设备视为文件。

SPOOLing技术

脱机：设置外围机，CPU只负责当需要输入数据时从磁盘中获取，当输出数据时输出到磁盘中；通过外围机来实现数据由磁盘到输出设备或者数据由输入设备到磁盘。

基于此，在计算机中模拟这个过程，成为假脱机。需要两个进程（预输入进程/缓输出进程）来模拟外围机，在磁盘中开辟两个井（输入井/输出井）来模拟输入设备和输出设备。

IO软件的分层思想：1、底层软件让高层软件具有独立与硬件的特性 2、高层软件提供良好接口

（1）用户进程层
执行I/O系统调用，对I/O数据进行格式化
（度算法：这一块可以参考进程调度算法进行，基本一致：1、fcfs 2、最短寻道时间优先 3、scan算法 4、Cscan算法 5、N-scan 6、F-scan