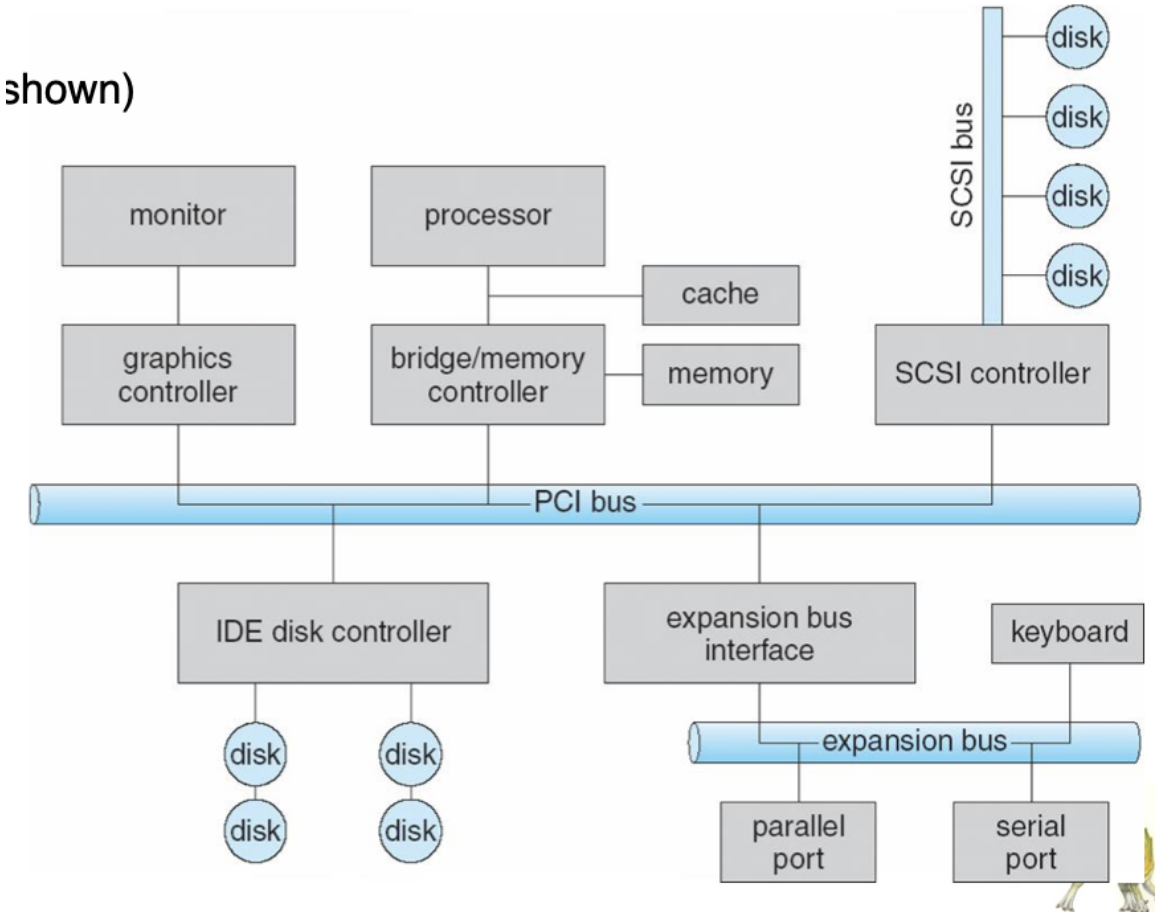
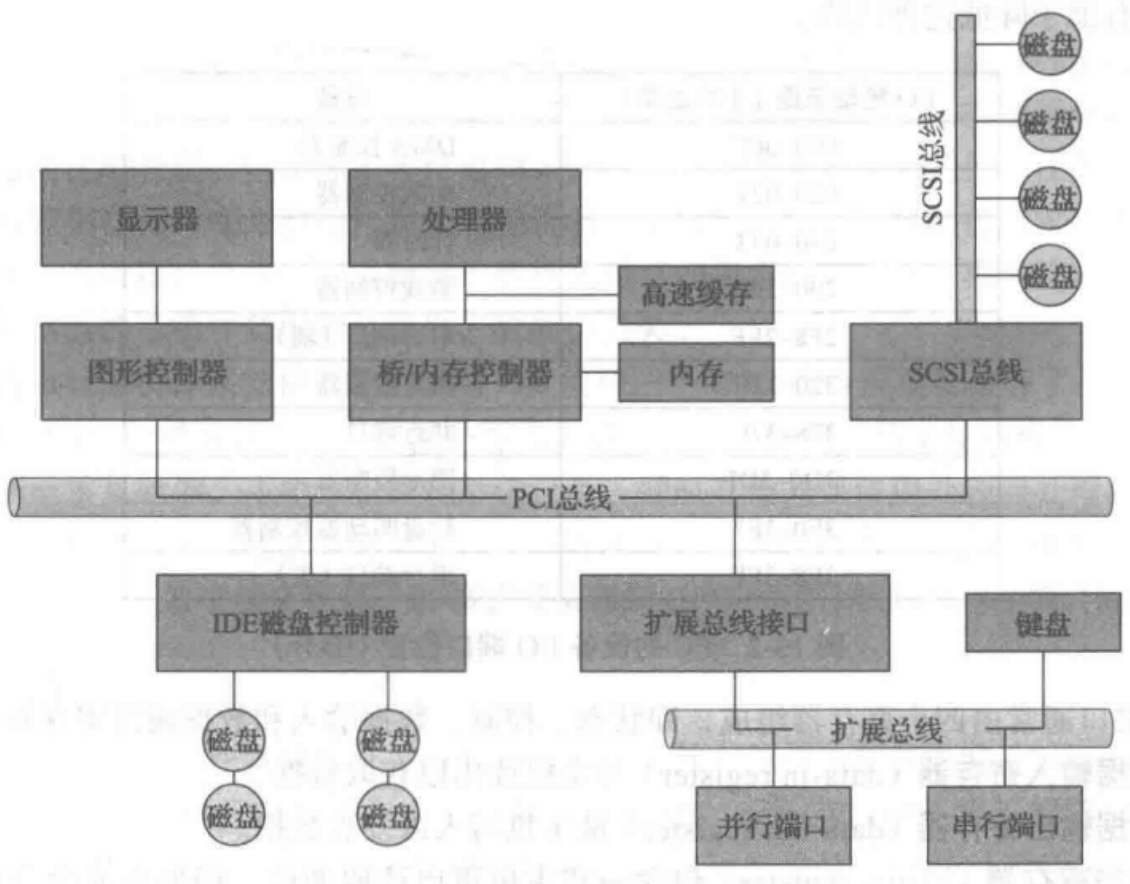
12 IO Systems

- Overview
  - I/O management is a major component of operating system designand operation
    - I/O管理是操作系统设计和运行的一个重要组成部分
    - ▪ Important aspect of computer operation
      - 电脑操作的重要方面
    - ▪ I/O devices vary greatly
      - ▪I/O设备变化很大
    - ▪ Various methods to control them
      - ▪控制它们的各种方法
    - ▪ Performance management
      - ▪绩效管理
    - ▪ New types of devices frequent
      - ▪新型设备频繁出现
  - io不同分类
    - Human-readable and machine-readable
      - Human-readable
        - 人类可读的
        - mouse, keyboard, and so on,
          - 鼠标，键盘等等，
      - machine-readable
        - 机器可读的
        - sensors, controllers, disks, etc.
          - 传感器、控制器、磁盘等
    - Transfer of data
      - character-oriented device - accepts and delivers the data as a stream of characters/bytes
        - 面向字符的设备——作为字符/字节流接受并传送数据
      - block-oriented device – accepts and delivers the data as fixed-size blocks
        - 面向块的设备——以固定大小的块接受并传送数据
    - Type of access
      - sequential device such as a tape drive.
        - 磁带机等顺序设备。
      - random access device, such as a disk.

        - 随机存取设备，如磁盘。
    - Network device - to send or receive data on a network.
        - 网络设备-在网络上发送或接收数据
- I/O Hardware
    - 图示





- port-设备与计算机的通信通过连接点或者端口
- bus-一组线路和通过线路传输信息的严格定义的协议,总线（Bus）是一种用于在计算机内部传输数据、地址和控制信号的通信系统
    - a bus is a communication system used to transfer data, addresses, and control signals between different components inside a computer or between multiple computers.
    - 菊花链-a到b到c,c到计算机
- Common concepts – signals from I/O devices interface with computer
    - 通用概念-来自I/O设备与计算机接口的信号
    - Port – connection point for device
        - 端口-设备连接点
    - ▪ Bus - daisy chain or shared direct access
        - 总线-菊花链或共享直接访问
        - PCI bus common in PCs and servers, PCI Express (PCIe)
            - 在pc和服务器中常见的PCI总线，PCI Express (PCIe)
        - ▪ expansion bus connects relatively slow devices
            - ▪扩展总线连接相对较慢的设备
    - ▪ Controller (host adapter) – electronics that operate port, bus, device
        - ▪控制器(主机适配器)-操作端口、总线、设备的电子设备
        - Sometimes integrated

        - 有时集成
    - Sometimes separate circuit board (host adapter)
        - 有时单独电路板(主机适配器)
    - Contains processor, microcode, private memory, bus controller, etc
        - 包含处理器，微码，私有存储器，总线控制器等
        - – A controller could have its own processor, memory, etc. (E.g.: SCSI controller)
            - -一个控制器可以有自己的处理器，内存等(例如:SCSI控制器)
- How can the processor give commands and data to a controller to accomplish an I/O transfer?
    - 处理器如何向控制器发出命令和数据以完成I/O传输?

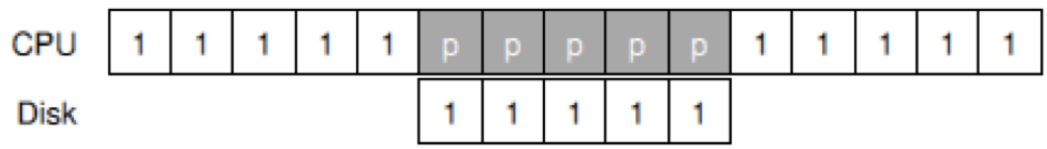
      处理器如何对控制器发出命令和数据以便完成 I/O 传输？简单答案是，控制器具有一个或多个寄存器，用于数据和控制信号。处理器通过读写这些寄存器的位模式来与控制器通信。这种通信的一种方式是，通过使用特殊 I/O 指令针对 I/O 端口地址传输一个字节或字。
    - Each controller has registers that are used for communicating with theCPU.
        - 每个控制器都有用于与cpu通信的寄存器。
    - There are data buffer that the operating system can read and write.
        - 有操作系统可以读写的数据缓冲区。
    - Each I/O port (device) is identified by a unique port address
        - 每个I/O端口(设备)由唯一的端口地址标识
    - Each I/O port consists of 4 registers (1 to 4 bytes in size) :
        - 每个I/O端口由4个寄存器(大小为1到4字节)组成:
        - data registers to pass data to the device or get data from
            - 将数据传递到设备或从中获取数据的数据寄存器
            - data-in register
                - 数据输入寄存器
            - data-out register
                - 输出数据寄存器
        - status register - can be read to see the current status of the device.
            - 状态寄存器-可以读取以查看设备的当前状态。
        - control register - to tell the device to perform a certain task.
            - 控制寄存器——告诉设备执行某项任务
    - 不同的读写寄存器的位模式来与控制器通信
        - Port-mapped I/O
            - Use different address space from memory
                - 使用不同于内存的地址空间
            - Access by special I/O instruction (e.g. IN, OUT)

- 通过特殊I/O指令(例如IN, OUT)进行访问,io触发总线线路,选择适当设备
- Memory-mapped I/O
  - 内存映射I / O
  - 设备的寄存器被映射到处理器的地址空间
  - Reserve specific memory space for device
    - 为设备预留特定的内存空间
  - Access by standard data-transfer instruction (e.g. MOV)
    - 通过标准数据传输指令(如MOV)进行访问
    - o More efficient for large memory I/O (e.g. graphic card)
      - o更有效的大内存I/ o(如图形卡)
    - o Vulnerable to accidental modification, error
      - o易发生意外修改、错误

- CPU – I/O interaction
  - I/O communication techniques /
    - Three techniques by which an I/O operation can be performed on a device.
      - 可以在设备上执行I/O操作的三种技术。
    - These are known as I/O communication techniques.
      - 这些被称为I/O通信技术。
    - Techniques to interact with a device and control the transfer with a device:
      - 与设备交互和控制设备传输的技术;
      - Polling (Programmed I/O)
        - 轮询(已编程I/O)
      - Interrupt-driven I/O
        - 中断驱动I / O
      - Direct Memory Access
        - 直接存储器访问
  - Polling
    - The data transfer is initiated by the instructions written in a computer program.
      - 数据传输是由写在计算机程序中的指令发起的。
    - CPU executes a busy-wait loop – periodically checking status of the device to see if it is time for the next I/O operation (tests the channel status bit).
      - CPU执行一个忙碌等待循环——定期检查设备的状态，看看是否到了下一个I/O操作的时间(测试通道状态位)。
    - CPU stays in a loop until the I/O device indicates that it is ready for data transfer.
      - CPU一直处于循环状态，直到I/O设备指示它准备好进行数据传输。

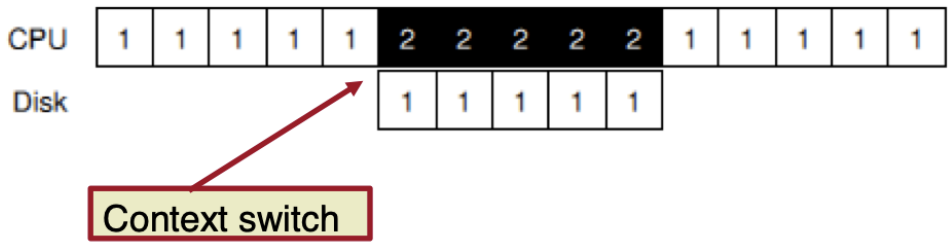
1. 主机重复读取忙位，直到该位清零。
2. 主机设置命令寄存器的写位，并写出一个字节到数据输出寄存器。
3. 主机设置命令就绪位。
4. 当控制器注意到命令就绪位已设置，则设置忙位。
5. 控制器读取命令寄存器，并看到写命令。它从数据输出寄存器中读取一个字节，并向设备执行 I/O 操作。
6. 控制器清除命令就绪位，清除状态寄存器的故障位表示设备 I/O 成功，清除忙位表示完成。

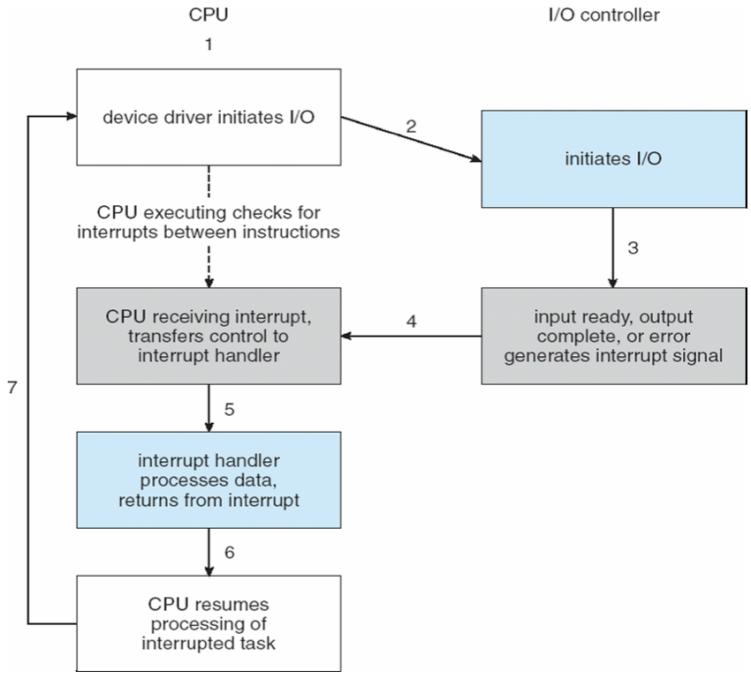
- Polling can be very fast and efficient, if both the device and the controller are fast and if there is significant data to transfer.
  - 如果设备和控制器都很快，并且有大量数据需要传输，那么轮询可以非常快速和有效。
- Disadvantage: It keeps the processor busy needlessly and leads to wastage of the CPU cycles
  - 缺点:它使处理器不必要地忙碌，导致CPU周期的浪费
    - 它将进入一个循环，等待I/O事件的发生，直到超时或至少一个事件就绪。这个过程需要消耗CPU资源来不断地检查文件描述符的状态。如果文件描述符一直没有就绪，那么这个循环会一直执行，浪费处理器的时间和能源
    - 一直读取状态寄存器,直到忙位被清除
-

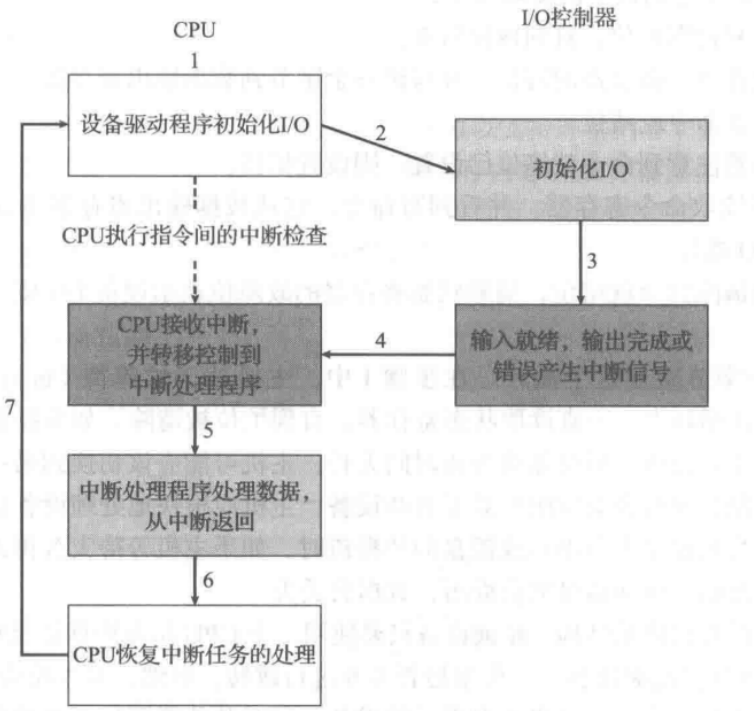


- Interrupts
  - In pooling the processor time is wasted =》 a hardware mechanism = interrupt.
    - 在polling中，处理器时间被浪费了=“硬件机制=中断”。
  - the CPU has an interrupt-request line that is sensed after every instruction.
    - CPU有一条中断请求线，在每条指令之后都会被感知。
  - Interrupts allow devices to notify the CPU when they have data to transfer or when an operation is complete. The CPU transfers control to the interrupt handler.
    - 中断允许设备在有数据要传输或操作完成时通知CPU。CPU将控制转移到中断处理程序。
  - Most CPUs have two interrupt-request lines:
    - 大多数cpu有两条中断请求行:
    - Non-maskable - for critical error conditions.
    - Maskable - used by device controllers to request / the CPU can temporarily ignore during critical processing.
      - 不可屏蔽-用于严重错误条件。

    - 可屏蔽的——设备控制器用来请求/ CPU在关键处理期间可以暂时忽略。
- How does the processor know when the I/O is complete?
    - 处理器如何知道I/O何时完成?
    - Through an interrupt mechanism (see Interrupt-driven I/O Cycle).- when the operation is complete, the device controller generates an interrupt to the processor.
        - 通过中断机制(参见中断驱动I/O周期)。—当操作完成后，设备控制器将生成一个中断处理器。
    - NB: the processor checks for the interrupt after every instruction cycle.
        - 注意:处理器在每个指令周期后检查中断。
    - after detecting an interrupt, the processor will perform a context switch, by executing the appropriate Interrupt Service Routine.
        - 在检测到中断后，处理器将通过执行适当的中断服务例程来执行上下文切换。
    - Interrupt handler receives interrupts the processor then performs the data transfer for the I/O operation.
        - 中断处理程序接收中断，然后为I/O操作执行数据传输。
        -



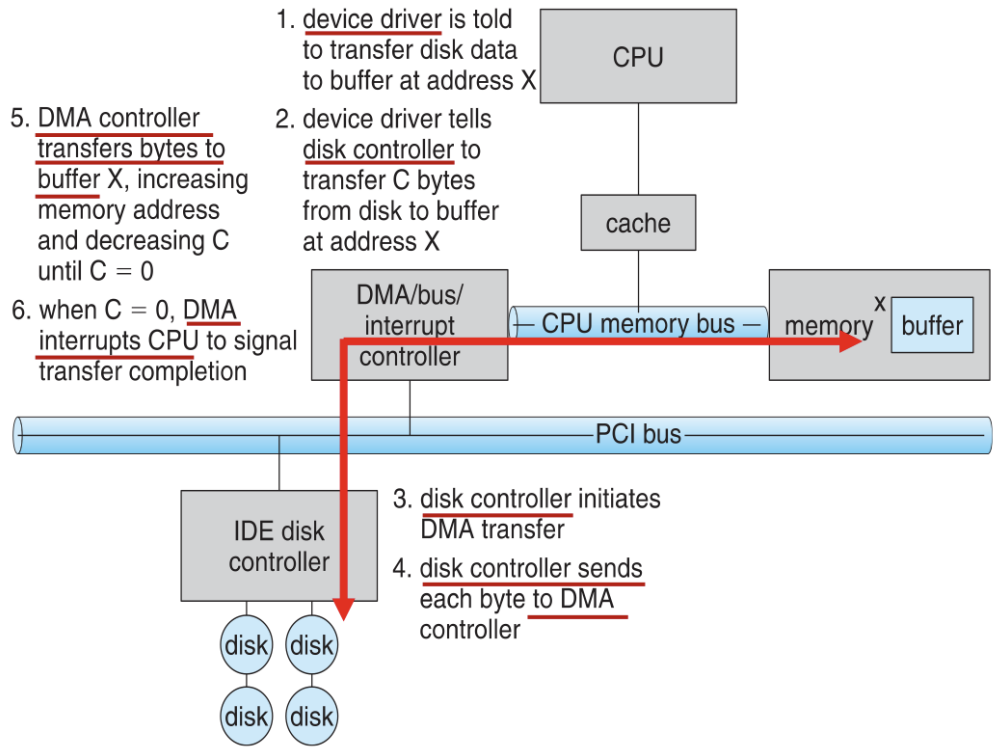


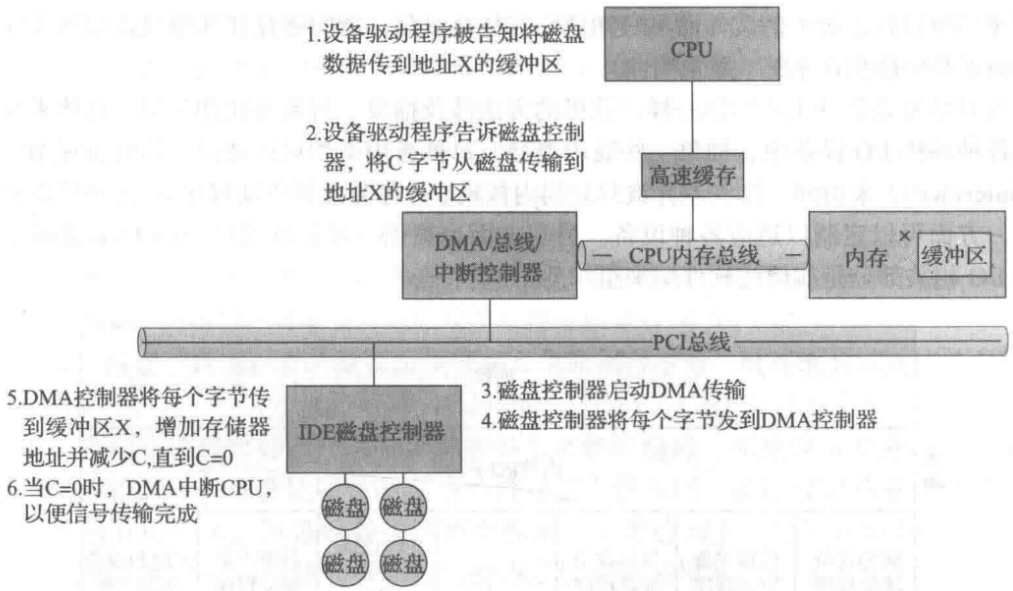


- Direct Memory Access
  - DMA = Direct Memory Access专用的控制器
    - when the data are large, interrupt driven I/O is not efficient.
      - 当数据量很大时，中断驱动的I/O效率不高。
    - instead of reading one character at a time through the processor, a block of characters is read at a time.
      - 不是通过处理器一次读取一个字符，而是一次读取一个字符块。
    - bypasses CPU to transfer data directly between I/O device and memory
      - 绕过CPU直接在I/O设备和内存之间传输数据
    - DMA hardware generates an interrupt when the I/O transaction is complete
      - 当I/O事务完成时，DMA硬件产生一个中断
    - requires DMA controller
      - 需要DMA控制器
    - version that is aware of virtual addresses can be even more efficient - Direct Virtual Memory Access DVMA
      - 直接虚拟内存访问DVMA,实现两个内存映射设备之间的传输,无需cpu和内存的干涉
  - To read or write a block, the processor sends the command to theDMA controller.
    - 要读或写一个块，处理器发送命令给dma控制器
  - The processor passes the following information to the DMA controller:
    - 处理器将以下信息传递给DMA控制器:
    - • The type of request (read or write)
      - •请求的类型(读或写)

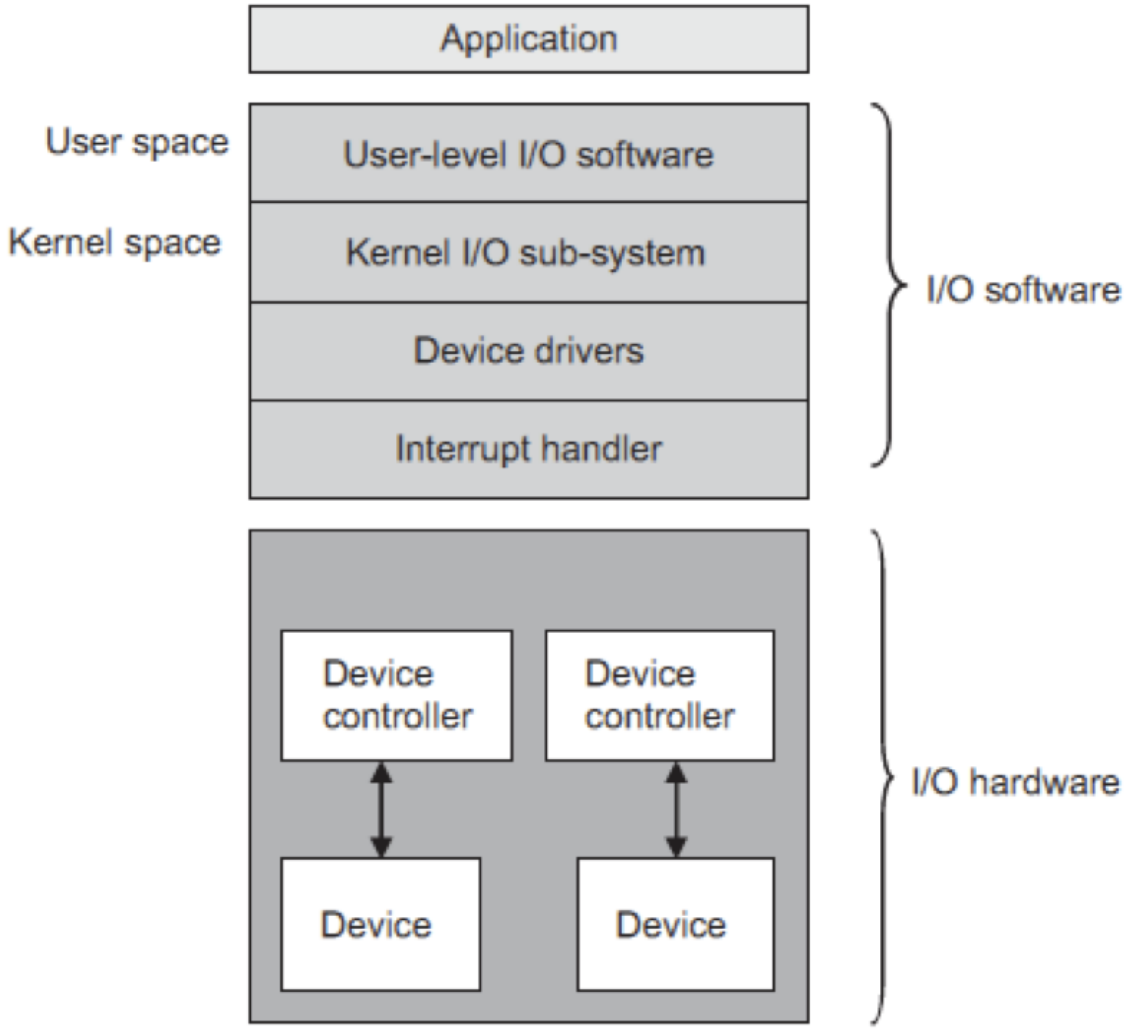
- • The address of the I/O device to which I/O operation is to becarried out
  - 要执行I/O操作的I/O设备的地址
- • The start address of the memory, where the data need to be written or read from, along with the total number of words.
  - •内存的起始地址，需要从哪里写入或读取数据，以及总字数。
- • The DMA controller then copies this address and the word count to its registers.
  - •DMA控制器然后复制这个地址和字数计数到它的寄存器

- 步骤
  -





- Layered structure of I/O
  -



- Application I/O Interface
  - User application access to a wide variety of different devices
    - 用户应用程序可以访问各种不同的设备
  - Characteristics of I/O devices
    -

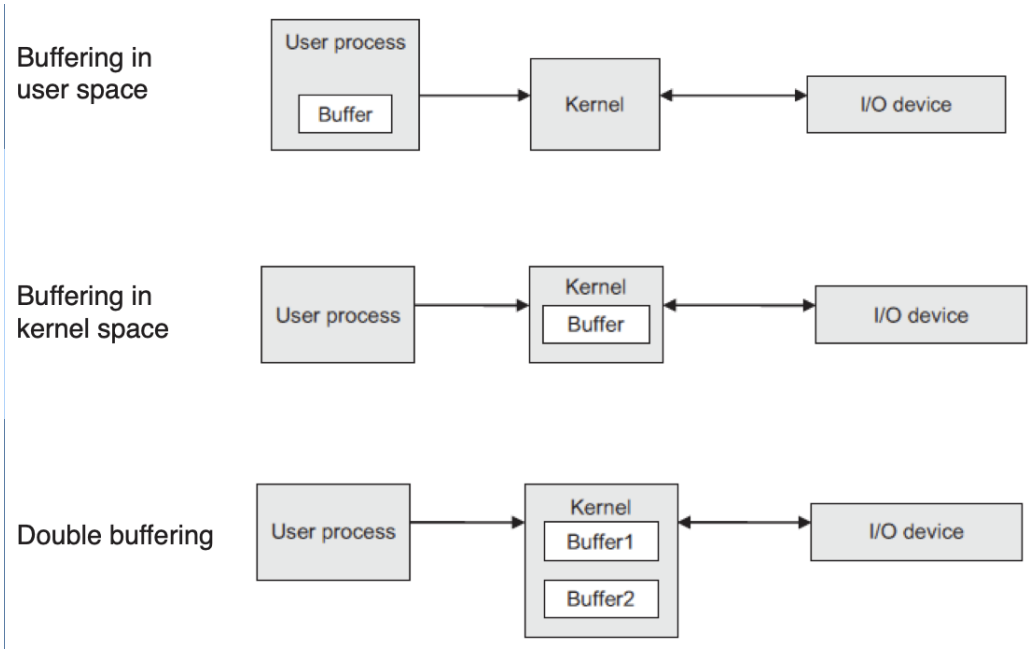
aspect	variation	example
data-transfer mode	character block	terminal disk
access method	sequential random	modem CD-ROM
transfer schedule	synchronous asynchronous	tape keyboard
sharing	dedicated sharable	tape keyboard
device speed	latency seek time transfer rate delay between operations	
I/O direction	read only write only read–write	CD-ROM graphics controller disk

方面	差异	例子
数据传输模式	字符，块	终端，磁盘
访问方式	顺序，随机	调制解调器，光盘
传输方式	同步，异步	磁带，键盘
分享	专用，共享	磁带，键盘
设备速度	延迟，寻道时间，传输速率，操作延迟	
I/O 方向	只读，只写，读写	光盘，图形控制器，磁盘

- Block and Character Devices
  - 块和字符设备
  - Block devices are accessed a block at a time
    - 块设备一次访问一个块
    - include disk drives and block-oriented devices
      - 包括磁盘驱动器和面向块的设备
  - commands include read, write, seek
    - 命令包括读、写、查找
    - Raw I/O (accessing blocks on a hard drive directly)
      - 原始I/O(直接访问硬盘上的块)
    - Direct I/O (uses the normal filesystem access)
      - 直接I/O(使用正常的文件系统访问)
    - Memory-mapped file I/O.
      - 内存映射文件I/O。
  - Character devices are accessed one byte at a time
    - 字符设备每次访问一个字节
    - include keyboards, mice, serial ports
      - 包括键盘、鼠标、串口
    - commands include get(), put()
      - 命令包括get()， put()
    - supported by higher-level library routines.
      - 由高级库例程支持。
- Network Devices
  - Varying enough from block and character to have own interface
    - 从块和字符变化足够有自己的界面
  - Linux, Unix, Windows and many others include socket interface
    - Linux、Unix、Windows和其他许多系统都包含套接字接口
    - socket acts like a cable or pipeline connecting two networked entities.
      - 套接字就像连接两个网络实体的电缆或管道。
  - Approaches vary widely (pipes, FIFOs, streams, queues, mailboxes)
    - 方法多种多样(管道、fifo、流、队列、邮箱)

- Clocks and Timers
  - Three types of time services are commonly needed in modernsystems:
    - 现代系统通常需要三种类型的报时服务:
    - o Get the current time of day.
      - 获取一天中的当前时间。
    - o Get the elapsed time since a previous event.
      - 获取自前一个事件以来经过的时间。
    - o Set a timer to trigger event X at time T.
      - 设置定时器，在时间T触发事件X。
  - A Programmable Interrupt Timer (PIT) is a hardware counter that generates an output signal when it reaches a programmed count.
    - 可编程中断定时器(PIT)是一种硬件计数器，当它达到可编程计数时产生输出信号。
- Blocking and Non-blocking I/O
  - Blocking – process/app is suspended (move it in the waiting queue) until I/O completed
    - 阻塞——进程/应用程序被挂起(将其移到等待队列中)，直到I/O完成
  - Nonblocking - the I/O request returns immediately, whether the requested I/O operation has (completely)occurred or not.
    - 非阻塞——I/O请求立即返回，无论请求的I/O操作是否(完全)发生。
    - Example:
      - ▪ a user interface that receives keyboard and mouse input while processing and displaying data on the screen.
        - ▪一个接收键盘和鼠标输入的用户界面，同时在屏幕上处理和显示数据。
      - ▪ a video application that reads frames from a file on disk while simultaneously decompressing and displaying the output on the display.
        - ▪一个视频应用程序，从磁盘上的文件读取帧，同时解压缩并在显示器上显示输出。
- Vectored I/O
  - known as scatter/gather I/O
    - 称为分散/聚集I/O
  - Scatter/gather refers to the process of gathering data from, or scattering data into, the given set of buffers.
    - 分散/收集是指从给定的缓冲区集合中收集数据或将数据分散到给定的缓冲区集合中的过程。
    - - read from or write to multiple buffers at once

        - -一次从多个缓冲区读取或写入
      - - allows one system call to perform multiple I/O operations
        - 允许一个系统调用执行多个I/O操作
      - - For example, Unix readve() accepts a vector of multiplebuffers to read into or write from.
        - —例如，Unix的readve()接受一个包含多个缓冲区的向量来读写。
- Kernel I/O Subsystem
  - Uniform Interface
    - there are different device drivers for various devices.UI makes uniform the interface, such that all the drivers' interface through a common interface.
      - 不同的设备有不同的设备驱动程序。UI使接口统一，使所有驱动程序的接口通过一个共同的接口。
  - Scheduling
    - Some I/O request ordering via per-device queue
      - 一些I/O请求通过每个设备队列排序
    - Some OSs try fairness
      - 一些尝试公平
  - Buffering
    - - store data in memory while transferring between devices
      - 缓冲-在设备之间传输时将数据存储在内存中
    - a buffer is an area where the data, being read or written, are copied in it, sothat the operation on the device can be performed with its own speed.
      - 缓冲区是一个区域，读取或写入的数据在其中被复制，以便设备上的操作可以以自己的速度执行。
      - Single buffering
        - kernel and user buffers
          - 内核和用户缓冲区
        - varying sizes
          - 不同大小
      - Double buffering – two copies of the data
        - 双重缓冲——数据的两个副本
        - permits one set of data to be used while another is collected.
          - 允许在收集另一组数据时使用一组数据
    -



- Caching
  - region of fast memory holding copy of data
    - 保存数据副本的快速存储器区域
  - involves keeping a copy of data in a faster-access location than where the data is normally stored.
    - 包括将数据的副本保存在比通常存储数据更快访问的位置。
  - Buffering and caching use often the same storage space
    - 缓冲和缓存通常使用相同的存储空间
  - cache 和 buffer
    - Cache（高速缓存）是一种存储器，它位于CPU和主存之间，用于加速CPU对内存的访问。Cache存储最近使用的数据和指令，以便CPU可以更快地访问它们，从而提高系统的性能。Cache使用一种快速的查找算法，可以快速地查找CPU需要的数据或指令，因此它可以大大减少CPU访问内存的次数，从而提高系统性能。
    - Buffer（缓冲区）是一种临时存储区域，它用于暂时存储数据，以便处理器或I/O设备可以在需要时进行访问。Buffer通常用于平衡处理器和I/O设备之间的速度差异，因为处理器和I/O设备之间的数据传输速度可能不一致。当处理器需要写入数据时，它可以将数据写入缓冲区，缓冲区可以在处理器和I/O设备之间缓存数据，以便处理器可以更快地继续执行其他操作，而不必等待I/O设备完成读写操作。
      - 作用不同：Cache主要用于加速CPU对内存的访问，Buffer主要用于平衡处理器和I/O设备之间的速度差异。
      - 存储位置不同：Cache位于CPU和主存之间，Buffer通常位于内存和I/O设备之间。
      - 存储内容不同：Cache存储最近使用的数据和指令，Buffer存储暂时的数据，用于处理器和I/O设备之间的数据传输。

        - 大小不同：Cache的大小通常比Buffer小得多，因为Cache需要更快的访问速度，需要使用更快的存储介质，而Buffer可以使用普通的内存作为存储介质。
        - 管理方式不同：Cache通常由CPU自动管理，而Buffer通常由操作系统或应用程序管理。
- Spooling and Device reservation
    - (SPOOL = Simultaneous Peripheral Operations On-Line) = buffers data for devices that cannot support interleaved data streams
        - (SPOOL =同步外设联机操作)=为不支持交错数据流的设备缓冲数据。
        - Spooling (Simultaneous Peripheral Operations On-line) is a technique used to share a single device (such as a printer) among multiple applications, to avoid resource conflicts and contention. In Spooling, all print jobs are first spooled to disk, forming a queue, and then output to the printer in a first-in-first-out (FIFO) order. This approach helps to avoid conflicts when multiple applications simultaneously request access to the printer, and can also improve printing efficiency.
            - Spooling（Simultaneous Peripheral Operations On-line）是一种技术，用于在多个应用程序之间共享同一设备（如打印机），以避免资源竞争和冲突。在Spooling中，所有的打印作业都先被缓存到磁盘中，形成一个作业队列，然后按照先进先出（FIFO）的顺序逐个输出到打印机。这种方式可以避免多个应用程序同时请求访问打印机而导致冲突的情况，同时也可以提高打印效率。
        - Device Reservation is a technique used to ensure that a device is exclusively reserved for the use of a single application at a given time. In device reservation, an application must request access to the device from the operating system before using it, to ensure that the device is not being used by another application during the requested time period. This approach helps to ensure that devices are fully utilized, and avoids resource conflicts and contention, while also improving system performance.
            - Device Reservation是一种技术，用于确保某个设备在某个时间只被一个应用程序独占使用。在设备预留中，应用程序在使用设备之前必须先向操作系统发出请求，以便保证设备在请求的时间段内不会被其他应用程序占用。这种方式可以确保设备被完全利用，避免资源竞争和冲突，同时也可以提高系统性能。
        - device can serve only one request at a time
            - 设备一次只能处理一个请求
        - spool queues can be general or specific.
            - 假脱机队列可以是通用的，也可以是特定的。
        - i.e., printers

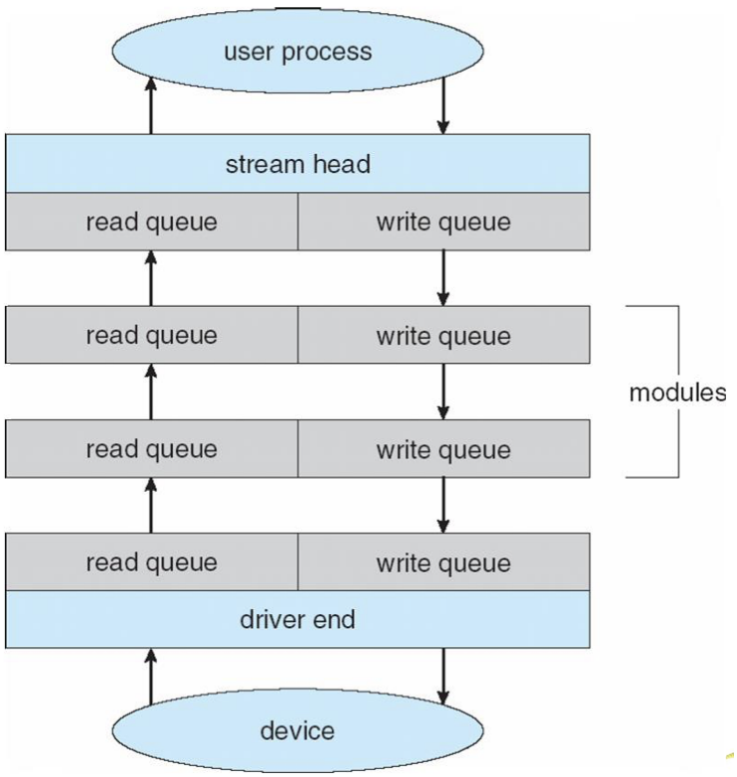
- 用于在多个应用程序之间共享同一设备（如打印机），以避免资源竞争和冲突。在Spooling中，所有的打印作业都先被缓存到磁盘中，形成一个作业队列，然后按照先进先出（FIFO）的顺序逐个输出到打印机。这种方式可以避免多个应用程序同时请求访问打印机而导致冲突的情况，同时也可以提高打印效率。
-

> 假脱机（spool）是保存设备输出的缓冲区，这些设备，如打印机，不能接收交叉的数据流。虽然打印机只能一次打印一个任务，但是多个应用程序可能希望并发打印输出，而不能让它们的输出混合在一起。操作系统通过拦截所有打印输出，来解决这一问题。应用程序的输出先是假脱机到一个单独的磁盘文件。当应用程序完成打印时，假脱机系统排序相应的假脱机文件，以便输出到打印机。假脱机系统一次一个地复制排队假脱机文件到打印机。对于有些操作系统，假脱机由系统守护进程来管理；对于其他，它由内核线程来处理。不管怎样，操作系统都提供了一个控制界面，以便用户和系统管理员显示队列，删除那些尚未打印的而不再需要的任务，当打印机工作时暂停打印，等等。
>
> 有些设备，如磁带机和打印机，无法实现复用多个并发应用程序的I/O请求。假脱机是，操作系统能够协调并发输出的一种方式。处理并发设备访问的另一种方法是提供明确的协调功能。有的操作系统（包括VMS）提供支持设备的互斥访问，以便允许进程分配一个空闲设备以及不再需要时释放设备。其他操作系统对这种设备的打开文件句柄有所限制。许多操作系统提供函数，允许进程协调互斥访问。例如，Windows提供系统调用来等待设备对象变得可用。系统调用`OpenFile()`也有一个参数，以便声明其他并发线程允许的访问类型。对于这些系统，应用程序需要自己来避免死锁。

- I/O protection - provides exclusive access to a device
  - I/O保护——提供对设备的独占访问
  - All I/O instructions defined to be privileged
    - 所有定义为特权的I/O指令
  - I/O must be performed via system calls that must be performed in kernel mode.
    - I/O必须通过必须在内核模式下执行的系统调用来执行。
  - Memory-mapped and I/O port memory locations must be protected by thememory management system.
    - (内存映射和I/O端口内存位置受内存管理系统保护。
- Error Handling – the I/O functions, when performed, may sometimes result in errors.
  - 错误处理——执行I/O函数时，有时可能会导致错误。
  - I/O requests can fail for many reasons, either momentary (buffers overflow) or permanent (disk crash).
    - I/O请求失败的原因有很多，可能是暂时的(缓冲区溢出)，也可能是永久性的(磁盘崩溃)。
  - Transient Errors - temporary reasons that cause any I/O processing to fail
    - 瞬态错误-导致任何I/O处理失败的临时原因
  - Permanent Errors - due to the failure of any device or wrong I/O request.
    - 永久错误-由于任何设备失败或错误的I/O请求。
- Device Driver
  - Piece of software a device driver knows in detail how a device works.

        - 设备驱动程序知道设备如何工作的详细信息。
    - The functions of a device driver are to:
        - 设备驱动程序的功能包括:
        - - accept the I/O requests from the kernel I/O sub-system.
            - -接受来自内核I/O子系统的I/O请求。
        - - control the I/O operation.
            - -控制I/O操作。
- Interrupt Handler
    - The device driver communicates with the device controllers, and then the device, with the help of the interrupt-handling mechanism.
        - 在中断处理机制的帮助下，设备驱动程序先与设备控制器通信，然后与设备通信。
    - The Interrupt Service Routine (ISR) is executed in order to handle a specific interrupt for an I/O operation.
        - 中断服务例程(ISR)是为了处理I/O操作的特定中断而执行的。
    - Transforming I/O Requests to Hardware Operations
        - 将I/O请求转换为硬件操作
        - Life Cycle of an I/O Request
            - I/O请求生命周期
- Streams
    - STREAM – a full-duplex communication channel between a user-level process and a device in Unix System V and beyond
        - 在Unix系统V和更高版本中，用户级进程和设备之间的全双工通信通道
    - A STREAM consists of:
        - The user process interacts with the stream head. User processes communicate with the stream head using either read( ) and write( )
            - 用户进程与流头交互。用户进程使用read()和write()与流头通信。
        - The device driver interacts with the device end.
            - 设备驱动程序与设备端交互。
        - zero or more stream modules between
            - 零或多个流模块之间
    - Each module contains a read queue and a write queue
        - 每个模块包含一个读队列和一个写队列
    - Message passing is used to communicate between queues
        - 消息传递用于在队列之间进行通信
        - Flow control can be optionally supported. Without flow control, data is passed along as soon as it is ready.

- 流量控制可以选择性地支持。如果没有流控制，数据一旦准备好就会被传递。



- Performance
  - Improving Performance
    - Reduce number of context switches
      - 减少上下文切换的次数
    - Reduce data copying
      - 减少数据复制
    - Reduce interrupts by using large transfers, smart controllers,polling
      - 通过使用大传输、智能控制器和轮询来减少中断
    - Use DMA
      - 使用直接存储器存取
    - Use smarter hardware devices
      - 使用更智能的硬件设备
    - Balance CPU, memory, bus, and I/O performance for highest throughput
      - 平衡CPU、内存、总线和I/O性能，以获得最高吞吐量