计算机组织结构

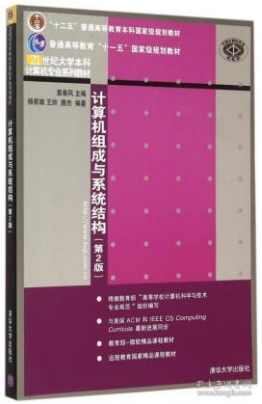
17 输入输出

刘博涵

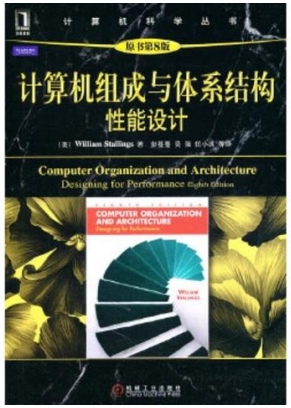
2023年12月21日

南京大学
NANJING UNIVERSITY

教材对应章节



第8章 互联及输入输出组织



第7章　输入/输出

你眼中的计算机是什么

外围设备

- 输入/输出操作通过连接到输入输出模块的各种外部设备完成，这些外部设备提供了在外部环境和计算机系统之间的数据交换，通常被称为外围设备（peripheral device），简称为外设（peripheral）
- 类型
  - 人可读设备：适用于与计算机用户通信
    - 显示器，打印机，……
  - 机器可读设备：适用于与设备通信
    - 磁盘，磁带，……
  - 通信设备：适用于与远程设备通信

能否将外设直接连接到系统总线上?

不可以

为什么不能把外设直接连接到系统总线上

- 外设种类繁多，操作方法多种多样
- 外设的数据传送速度一般比存储器或处理器的慢得多
- 某些外设的数据传送速度比存储器或处理器要快
- 外设使用的数据格式和字长度通常与处理器不同

I/O 模块

- 通过系统总线或中央交换器和存储器连接
- 通过专用数据线与一个或多个外设连接

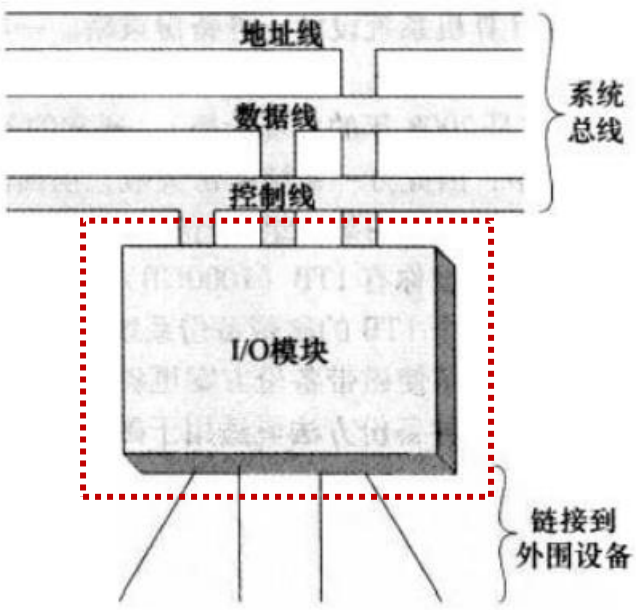
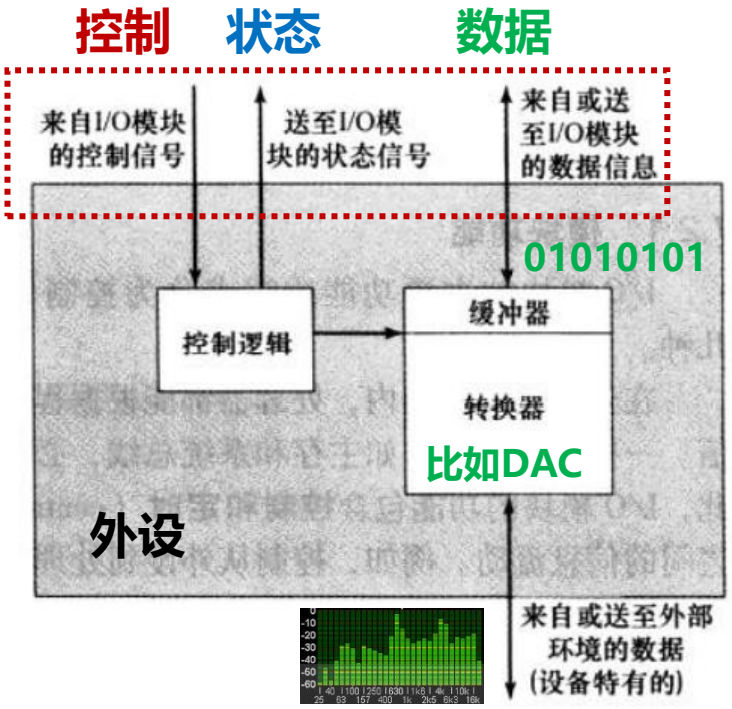


图 7-1　I/O 模块通用模型

I/O模块是计算机内部系统和外设之间的桥梁

外围设备的接口

- 输入/输出模块的接口以控制、状态和数据信号的形式出现
- 与设备相关的控制逻辑控制外设的操作，以响应来自输入/输出模块的命令
- 缓冲器用于缓存输入/输出模块和外设之间传送的数据
  - 缓冲器的大小一般为8位或16位

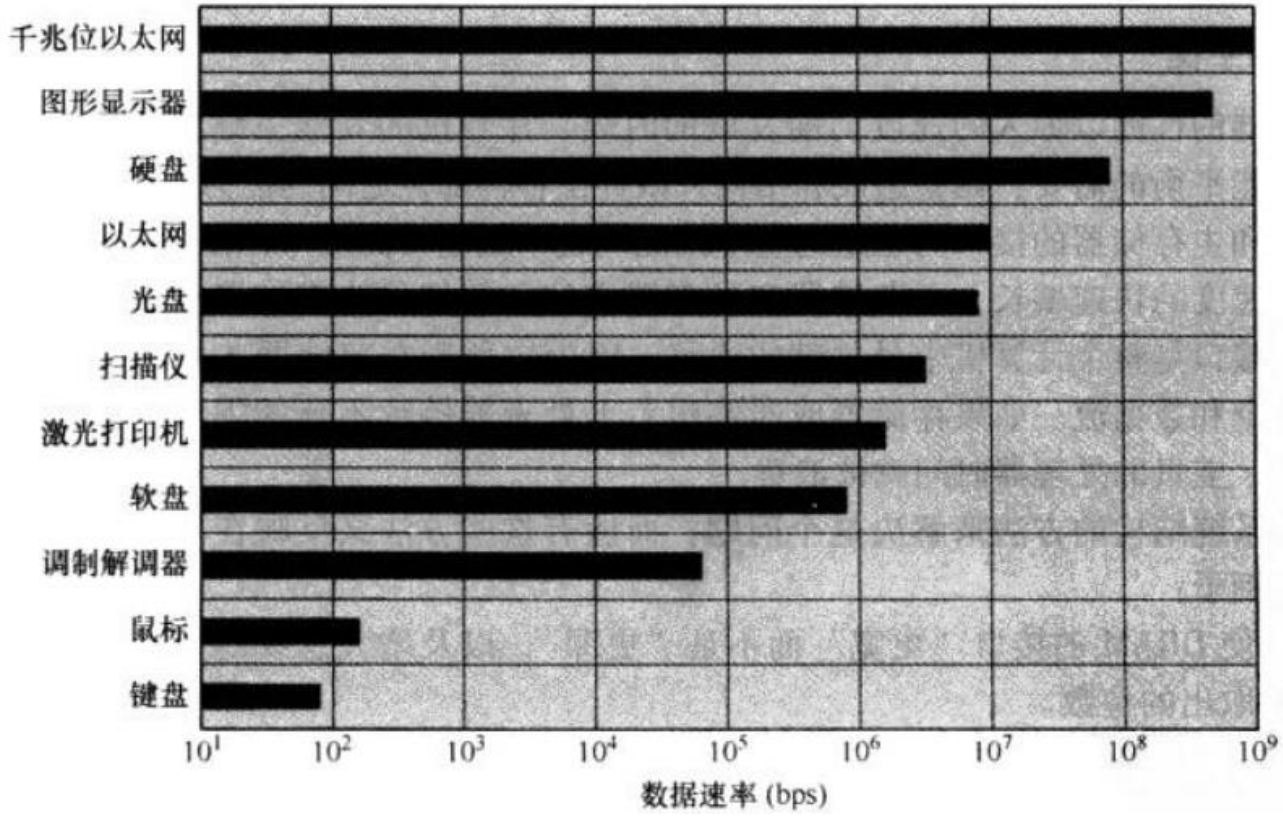


I/O模块的功能

- **处理器通信**
  - **命令译码**：输入/输出模块接收来自处理器的命令，这些命令一般作为信号发送到控制总线
  - **状态报告**：由于外设速度很慢，所以知道输入/输出模块的状态很重要
  - **数据**：数据是在处理器和输入/输出模块之间经由数据总线来交换的
  - **地址识别**：输入/输出模块必须能识别它所控制的每个外设的唯一地址
- **设备通信**
  - 通信内容包含**命令**、**状态**信息和**数据**

I/O模块的功能（续）

- **数据缓冲**
  - 外设的数据传送速度一般比存储器或处理器的慢得多
  - 某些外设的数据传送速度比存储器或处理器要快



I/O模块的功能（续）

- **控制和定时**
  - 处理器会非预期的与一个或几个外设进行通信
  - 一些内部资源，如主存和系统总线，是被共享的

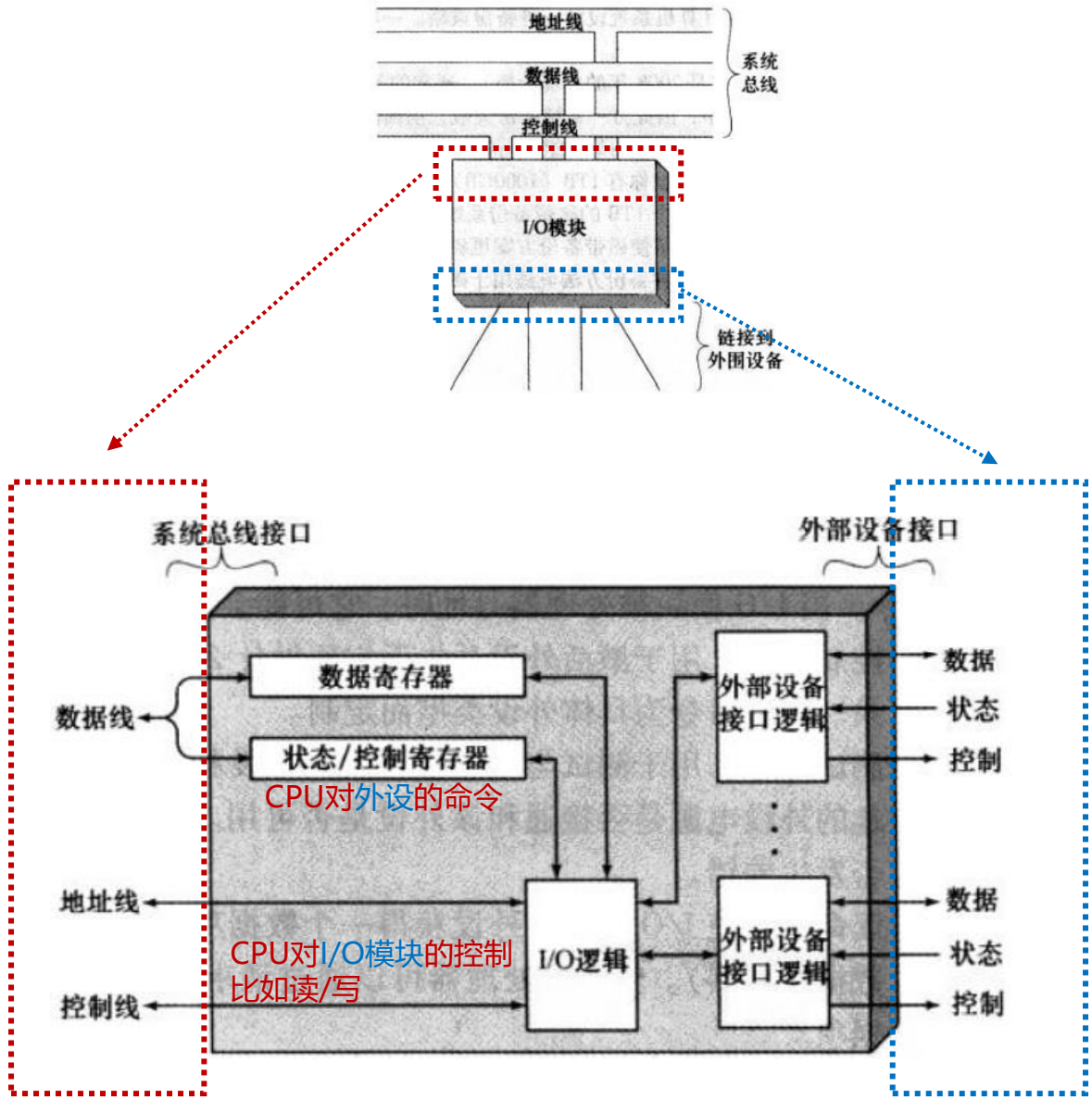
  } **原因**

  - 例如：控制从外设到处理器的数据传送包括以下几个步骤
    - 处理器**查询**输入/输出模块以检验所连接设备的**状态**
    - 输入输出模块**返回**设备**状态**
    - 如果设备运转并准备就绪，则处理器通过向输入/输出模块**发出**一条**命令**,请求数据传送
    - 经输入/输出模块**传送数据**到处理器

I/O模块的功能（续）

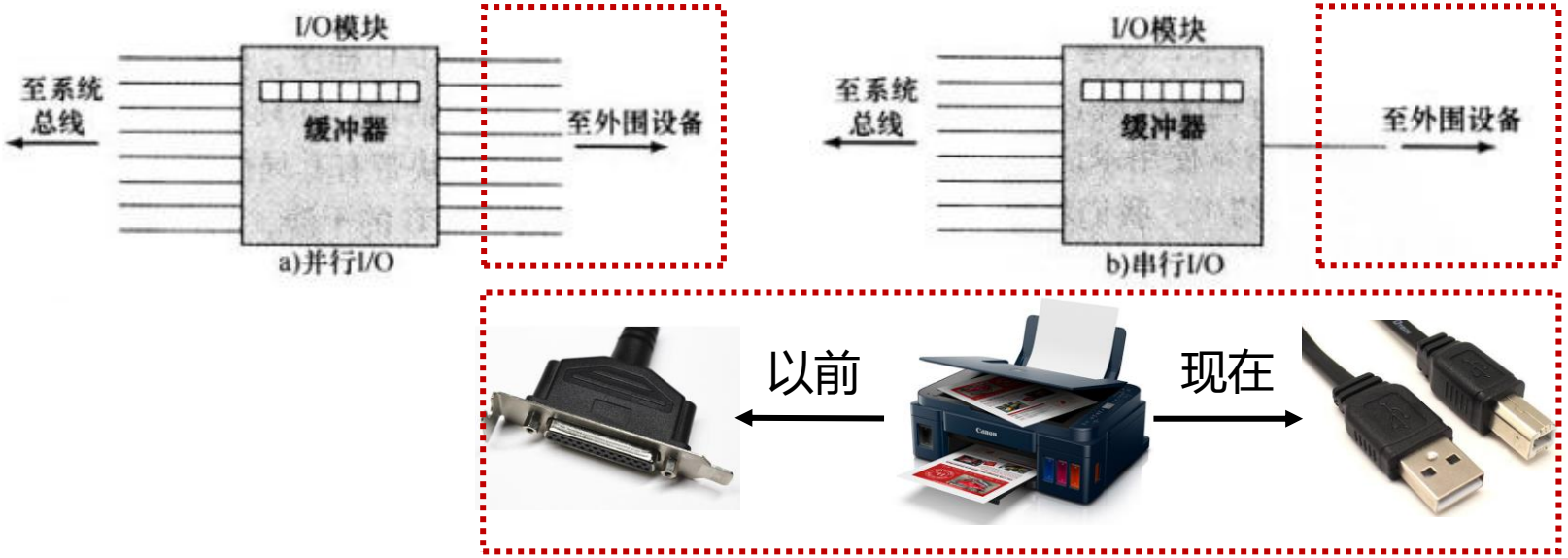
- **检错**
  - 检错并把差错信息报告给处理器
  - 差错类型
    - 设备报告的机械和电路故障
    - 传输过程中数据位的变化

I/O模块的结构


地址线
数据线
控制线
系统总线
I/O模块
链接到外围设备
系统总线接口
外部设备接口
数据寄存器
状态/控制寄存器
数据线
CPU对外设的命令
地址线
控制线
CPU对I/O模块的控制
比如读/写
I/O逻辑
外部设备接口逻辑
外部设备接口逻辑
数据
状态
控制
数据
状态
控制

外部接口

- **接口类型**
  - **并行接口**：多根线连接输入/输出模块和外设，同时传送多位数据
  - **串行接口**：只有一根线用于传输数据，每次只传输一位数据
- 由于并行接口要求每次同时传送，当传输速度和总线长度增加时，总线的时钟频率会受到限制



FireWire和USB

- FireWire：IEEE标准1394串行总线
- USB：通用串行总线（Universal Serial Bus）
  - 控制传输：令牌包，数据包，握手包
  - 批量传输：令牌包，数据包，握手包
  - 同步传输：令牌包，数据包
  - 中断传输：令牌包，数据包，握手包

	USB				FireWire		
	1.1	2.0	3.0	3.1	400	800	3200
数据传输速率	12Mbps	480Mbps	5Gbps	10Gbps	400Mbps	800Mbps	3.2Gbps
最多连接设备数	127				63		
总线类型	串行						
结构	主从设备模式				点对点		

I/O操作技术

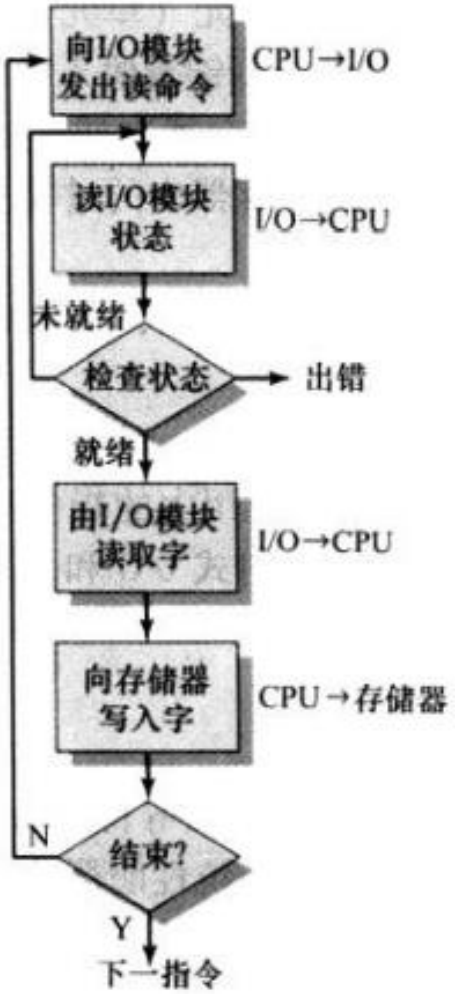
- **编程式 I/O**：处理器通过执行程序来直接控制I/O操作，当处理器发送一条命令到I/O模块时，它必须等待，直到I/O操作完成
- **中断驱动式 I/O**：处理器发送一条I/O命令后，继续执行其他指令；并且当I/O模块完成其工作后，才去中断处理器工作
- **直接存储器读取**（Direct Memory Access，DMA）：I/O模块与主存直接交换数据，而不需要处理器的干涉

传递方式	无中断	使用中断
I/O 与存储器之间的传递通过处理器实现	编程式 I/O	中断驱动式 I/O
I/O 与存储器直接传送		直接存储器存取（DMA）

编程式I/O

- 当处理器在执行过程中遇到一条与I/O操作有关的指令时，它通过发送指令到适当的I/O模块来执行这条指令
- I/O模块将执行所要求的动作，然后在I/O状态寄存器中设置一些适当的位
- I/O不会中断处理器，因此处理器需要周期性地检查I/O模块的状态，直到发现该操作完成

CPU全程100%投入



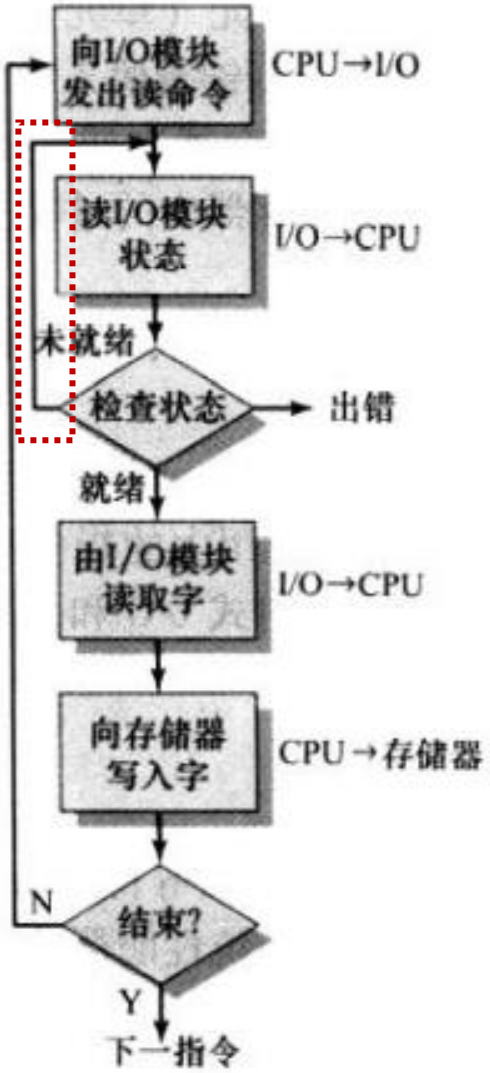
编程式I/O：I/O命令

- 为了执行I/O操作，处理器发送一个指定具体I/O模块和外设的地址，并发送一条**I/O命令**
- **类型**
  - **控制命令**：激活外设并告诉它要做什么
  - **测试命令**：测试I/O模块及其外设相关的各种状态条件
  - **读命令**：使I/O模块从外设获得一个数据，把它存入内部缓冲区
  - **写命令**：使I/O模块从数据总线获得一个数据，把它传入外设

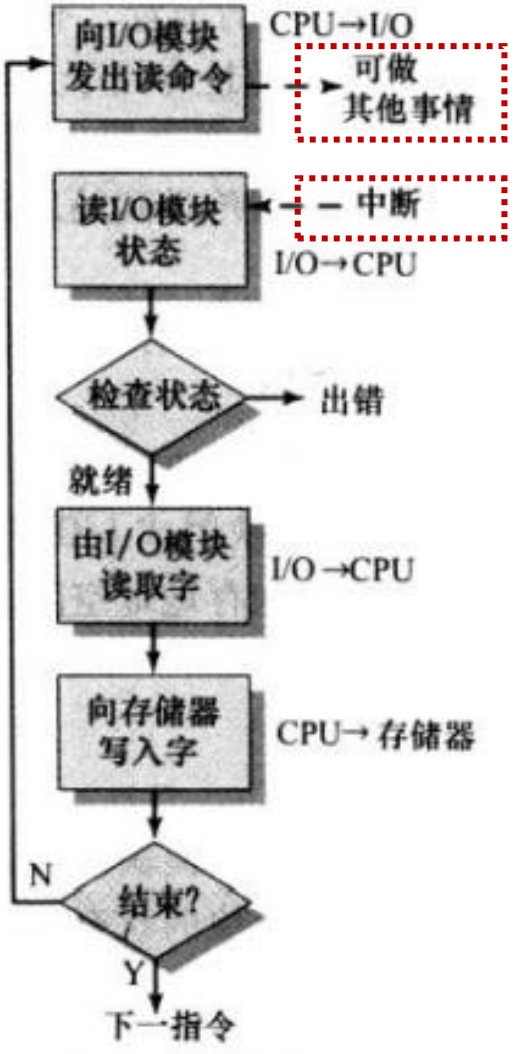
编程式I/O：I/O指令

- **I/O指令**很容易**映射**为**I/O命令**，并且两者之间通常是简单的一一对应关系
  - 指令的形式**取决于外设寻址的方式**
- **编址方式**
  - **存储器映射式I/O**：存储单元和I/O设备有统一的地址空间
    - 能使用大的指令系统，可进行更有效的编程
    - I/O设备占用地址空间
  - **分离式I/O**：让总线既有存储器的读线和写线，同时也有输入和输出命令线

中断驱动式I/O

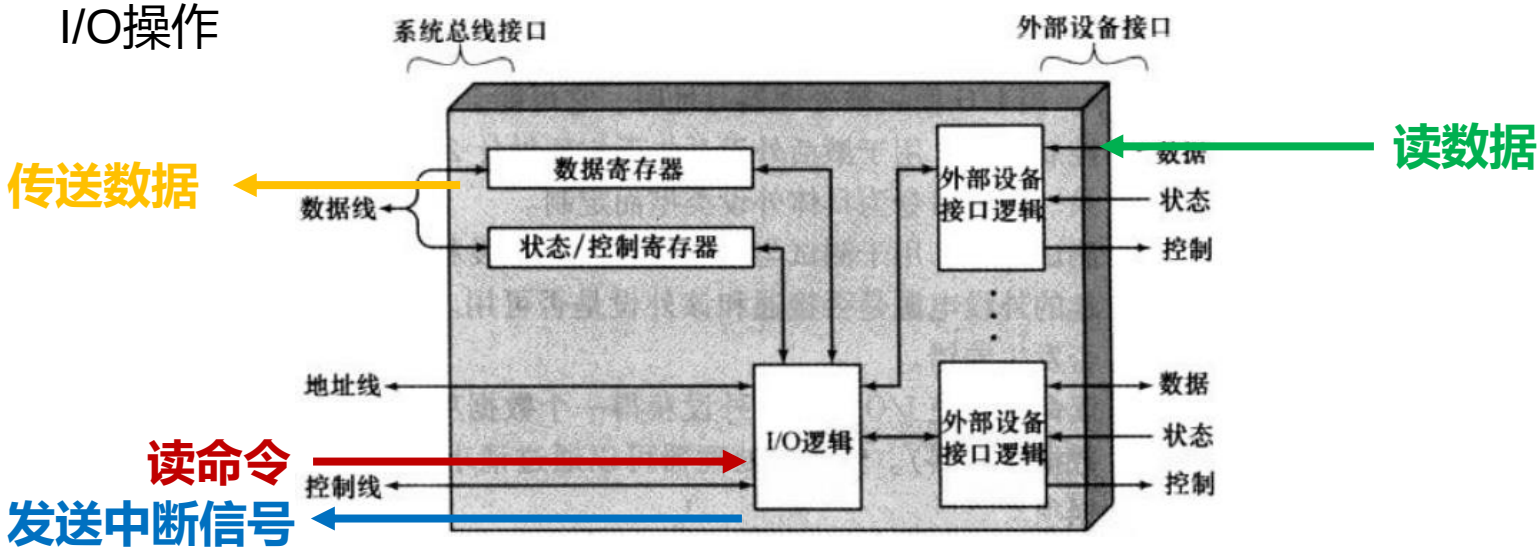


- 处理器发送一个I/O命令到模块，然后去处理其它有用的工作
- 当I/O模块准备和处理器交换数据时，它中断处理器以请求服务
- 处理器执行数据传送，最后恢复它原先的处理工作



中断驱动式I/O（续）

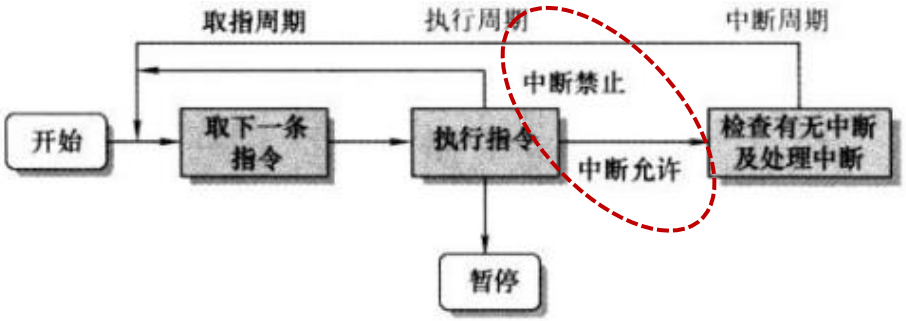
- **从I/O模块的角度来看**
  - I/O模块**接收**来自处理器的**读命令**
  - I/O模块从相关的外设中**读入数据**
  - 一旦数据进入I/O模块的数据寄存器后，该模块通过控制总线给处理器**发送中断信号**
  - I/O模块**等待**直到处理器请求该数据时为止
  - 当处理器有数据请求时，I/O模块把**数据传送**到数据总线上，并准备另一个I/O操作

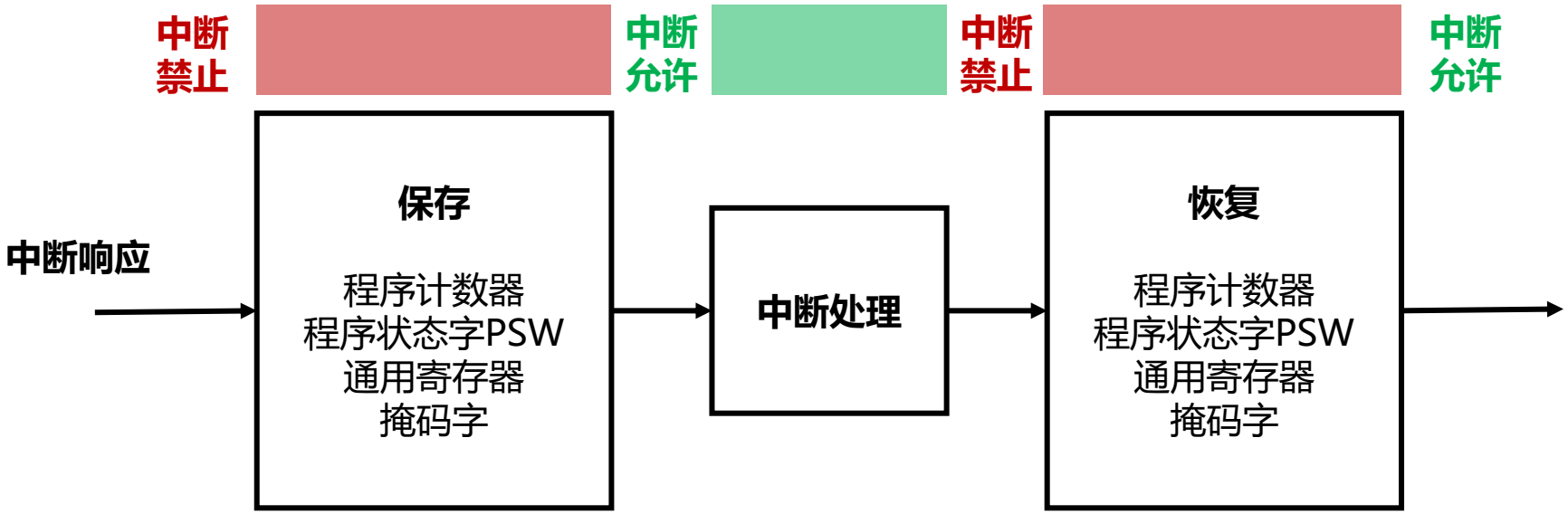


中断驱动式I/O（续）

- **从处理器的角度来看**
  - 处理器发送一个读命令
  - 处理器离开去做其它的事情，并在每个指令周期结束时检查中断
  - 当来自I/O模块的中断出现时，处理器保存当前程序的现场
  - 处理器从I/O模块读取数据字并保存到主存中
  - 处理器恢复刚才正在运行的程序的现场，并继续运行原来的程序

中断驱动式I/O：中断允许和中断禁止





中断驱动式I/O：响应优先级和处理优先级

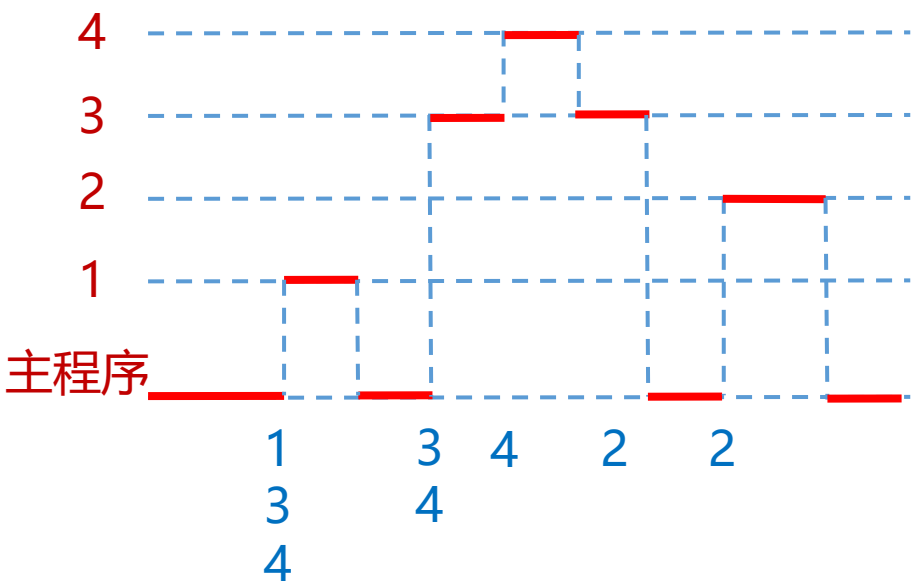
- **例子：**假设中断系统中有4个中断源，其**响应优先级**为L1>L2>L3>L4，**处理优先级**为L1>L4>L3>L2。如果在主程序执行时同时发生L1、L3和L4中断，并且在处理L3中断的过程中发生L2中断，写出**掩码字**和所有中断服务程序的过程。

掩码字 / 屏蔽字

	掩码字			
	L1	L2	L3	L4
L1	1	1	1	1
L2	0	1	0	0
L3	0	1	1	0
L4	0	1	1	1

行屏蔽列

中断服务程序



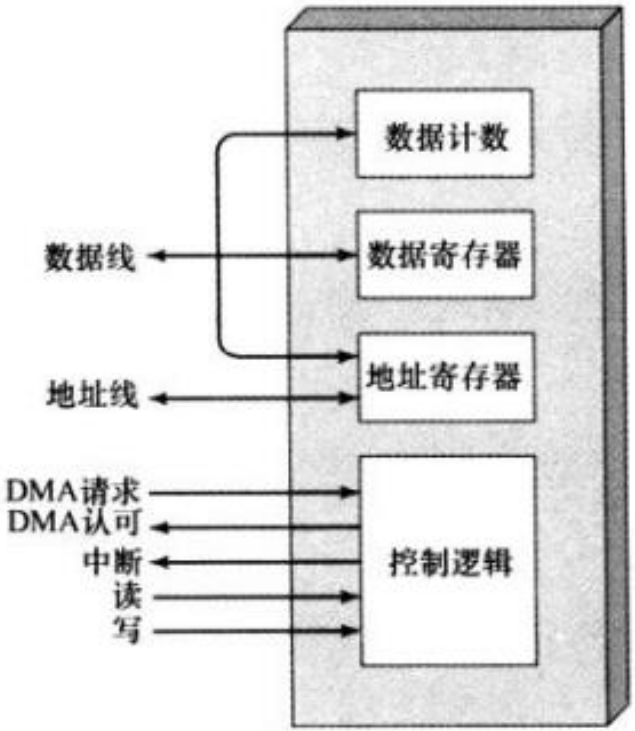
中断驱动式I/O：设备识别

- **多条中断线**：处理器仅仅挑选具有最高优先级的中断线
  - 即使有多条中断线可用，每条线上也需要采用其它三种技术中的一种
- **软件轮询**：模块的轮询次序就决定了模块的优先级
  - 轮询每一个I/O模块来确定是哪个模块发生的中断
- **菊花链**：链接模块次序就决定了模块的优先级
  - 所有的I/O模块共享一条中断请求线，中断应答线采用菊花链穿过这些中断模块
- **独立请求**：中断控制器决定
  - 特定的中断控制器用于解码和分析优先级

直接存储器存取（DMA）

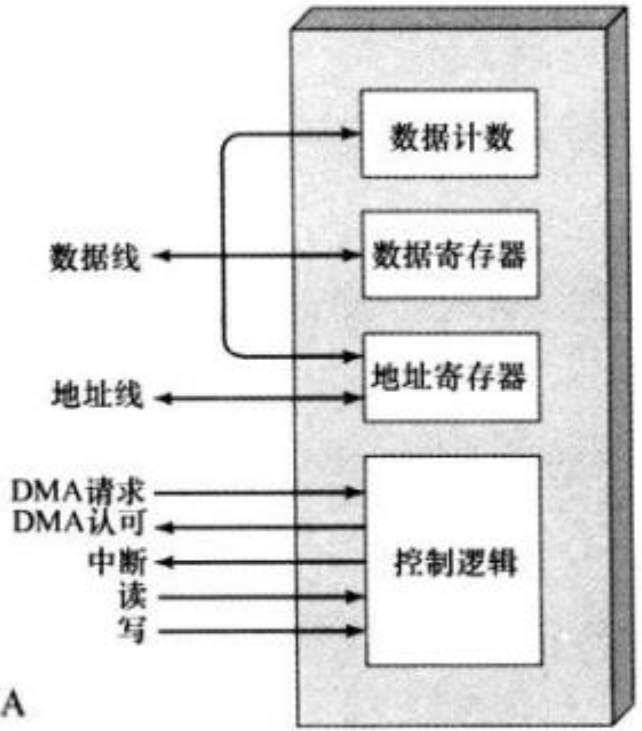
- **前两种技术的不足**
  - I/O传送速度受处理器测试和服务设备速度的限制
  - 处理器负责管理I/O传送，对于每一次I/O传送，处理器必须执行很多指令
- **直接存储器存取**
  - 无需经过处理器即可直接访问内存的模块

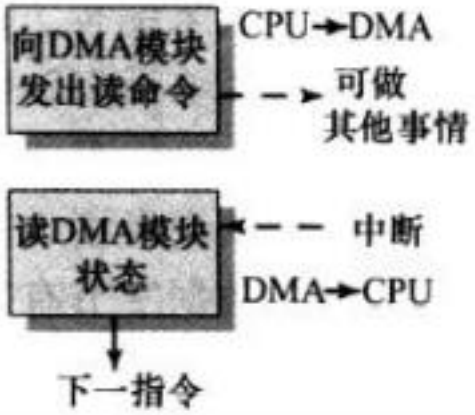
Intel 8237A



直接存储器存取（DMA）

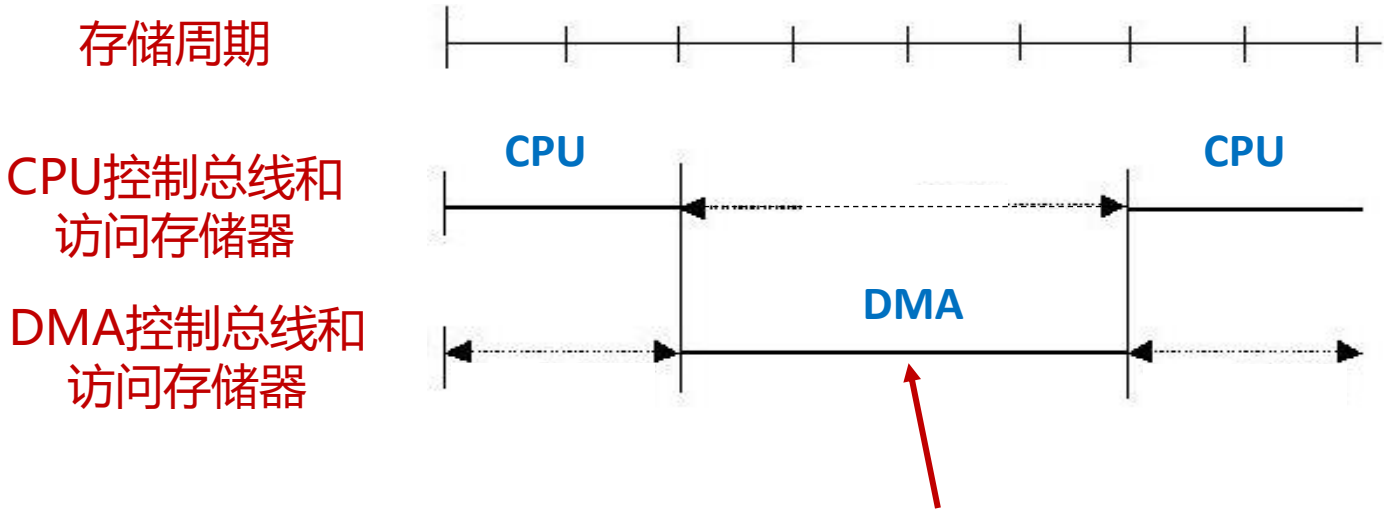
- 处理器通过发送以下信息向DMA模块发出命令：读/写、I/O设备地址、内存中的起始位置、字数
- 处理器继续进行其他工作
- DMA模块将全部数据块，每次一个字，直接将数据传输到存储器或从存储器读出，而无需经过处理器
- 当传输完成时，DMA模块向处理器发送一个中断信号





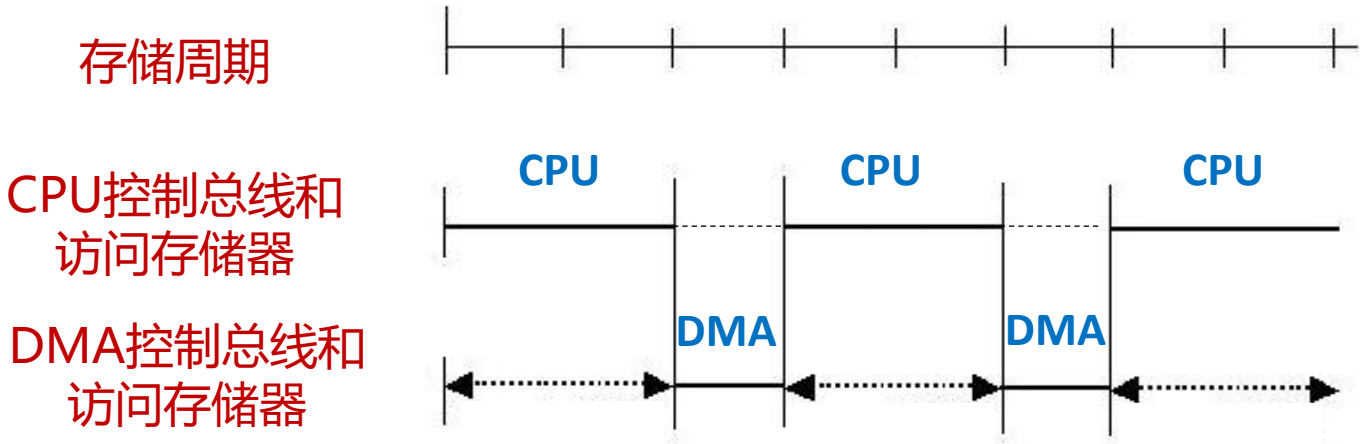
内存访问时，DMA优先于CPU

DMA内存访问：CPU停止法



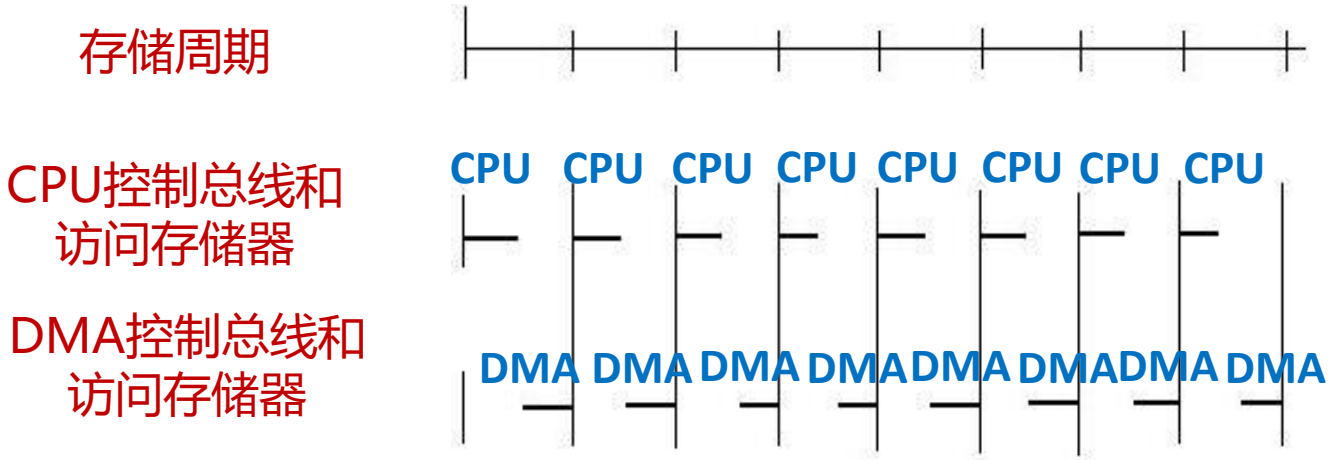
- **优点**：控制简单
- **缺点**：影响CPU，没有充分利用内存
- **适用**：高速I/O设备的块传输

DMA内存访问：周期窃取



- **优点**：充分利用CPU和内存，及时响应I/O请求
- **缺点**：DMA每次都请求总线
- **适用**：I/O周期大于存储周期

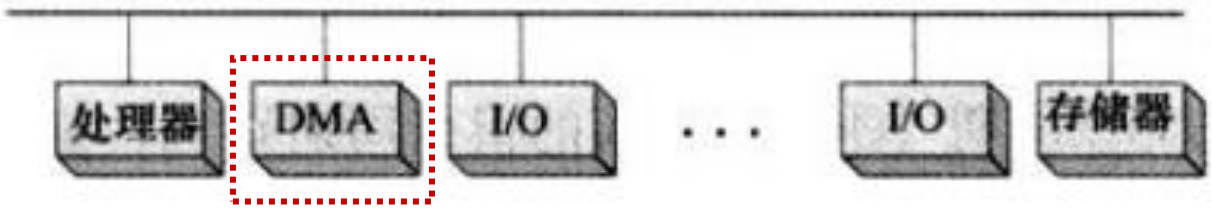
DMA内存访问：交替分时访问



- **优点**：CPU未停止或等待，DMA不请求总线
- **缺点**：CPU周期大于存储周期

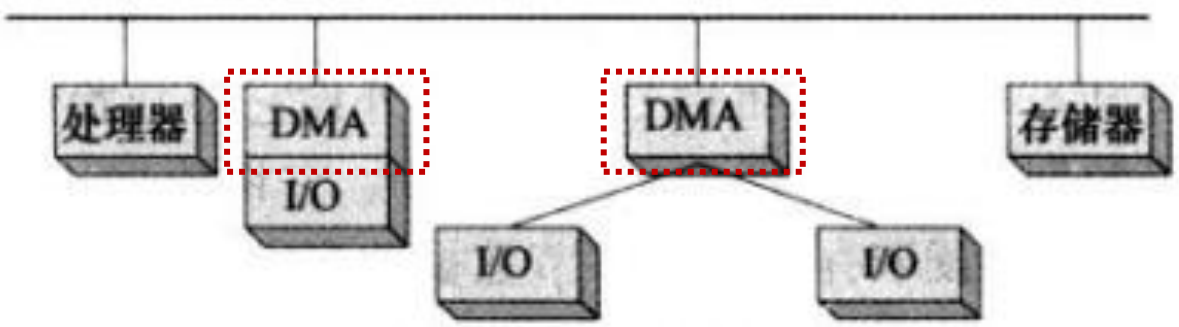
DMA配置机制：单总线分离DMA

- 所有模块**共享**相同的**系统总线**
- DMA模块使用编程式I/O，通过DMA模块在存储器和I/O模块之间交换数据
- 便宜但低效



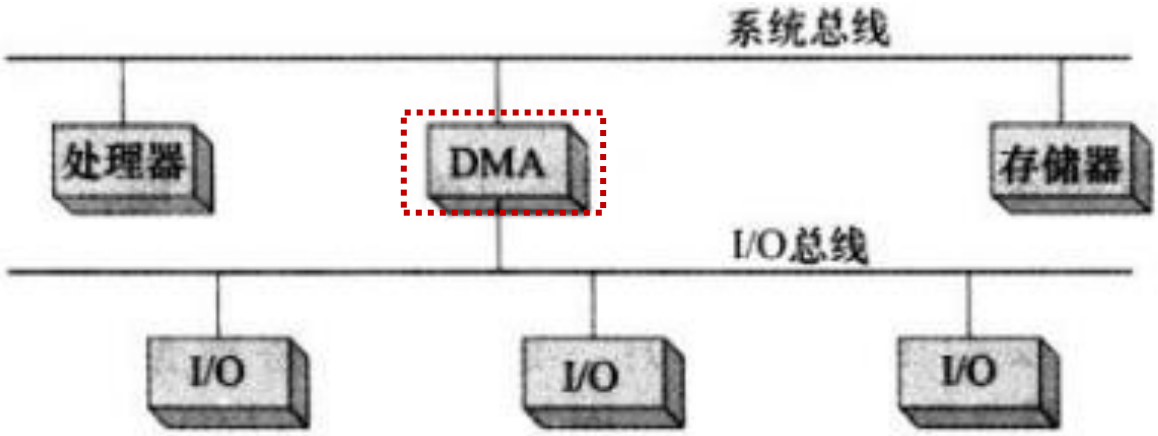
DMA配置机制：单总线集合的DMA-I/O

- DMA逻辑实际上可能**是I/O模块的一部分**，也可能是控制一个或多个I/O模块的单独模块
- 减少总线周期数

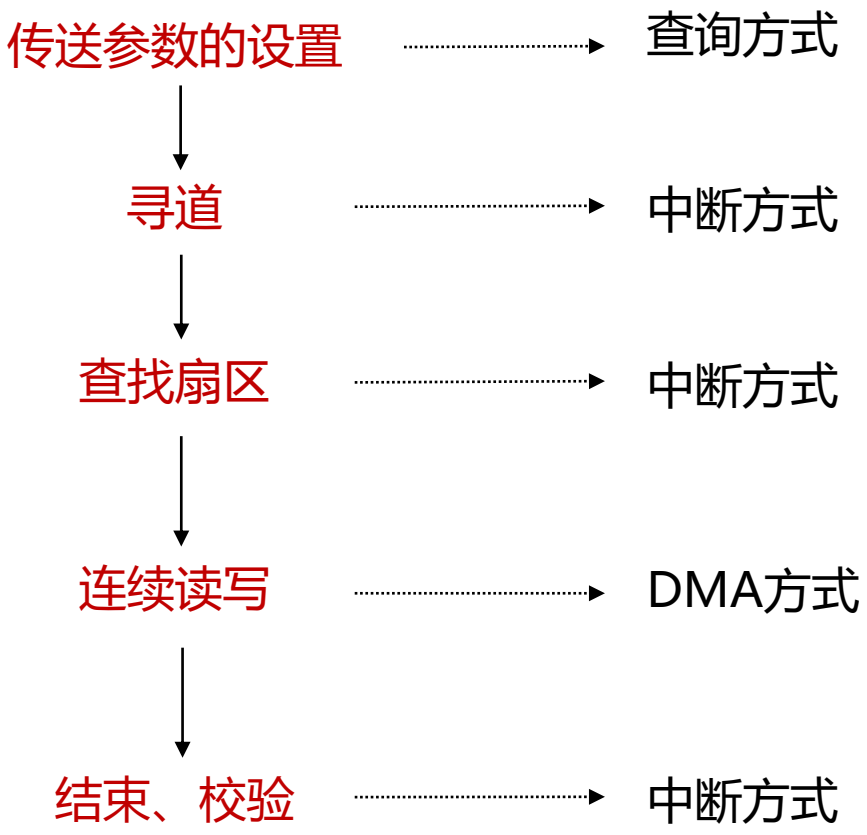


DMA配置机制：I/O 总线

- 使用I/O总线将I/O模块连接到DMA模块
- 多个I/O模块**共享DMA**，且易于扩展



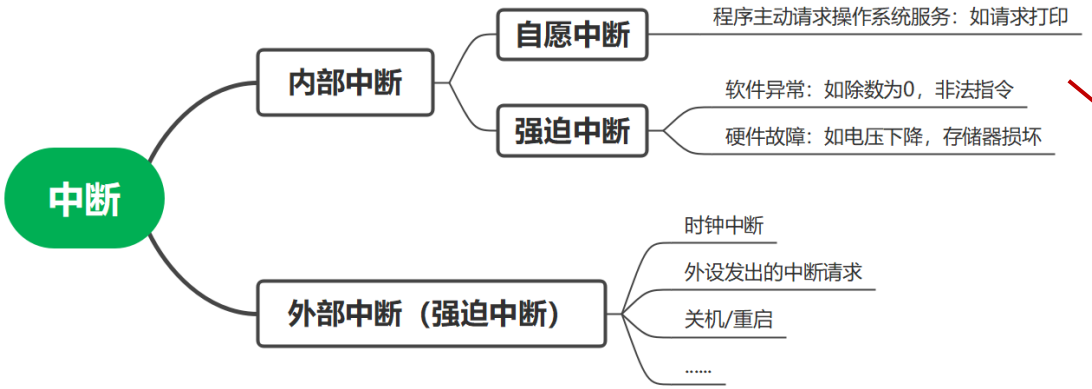
DMA示例：硬盘存取



I/O模块的演变

- CPU直接控制外设
- 增加控制器或I/O模块，CPU使用编程式I/O，将CPU与外围设备的细节分离
- 采用中断，CPU无需花费时间等待外围设备就绪
- I/O模块可通过DMA直接存取存储器，无需CPU负责存储器和I/O模块之间的数据传递
- I/O通道（I/O channel）：I/O模块有自己的处理器，带有专门为I/O操作定制的指令集
  - CPU指示I/O通道执行存储器中的I/O指令，只有在执行完成后才会中断CPU
- I/O处理器（I/O processor）：I/O模块有一个局部存储器，I/O模块成为一个自治的计算机，常用于与交互式终端进行通信
  - 只需最少的CPU参与即可控制大量I/O设备

补充阅读：中断类型和中断源识别



x86的内部中断	类型数
供用户定义的中断	224
保留的中断	27
专用的中断	5

- **中断源识别**
  - **CPU轮询**：所有设备共用一条中断请求线，简单但效率低
  - **原因寄存器**：MIPS采用此方法，处理器执行过程中硬件自动设置
  - **中断向量**：x86采用此方法，查询内存专用区域的中断向量表（中断服务程序的入口）

总结

- **外围设备**
- **I/O模块：功能，结构**
- **I/O操作技术**
  - 编程式I/O
  - 中断驱动式I/O
  - 直接存储器存取
- **I/O模块的演变**

谢谢

bohanliu@nju.edu.cn

南京大學
NANJING UNIVERSITY