第 3 章
输入 / 输出系统

郑宏 副教授
计算机学院
北京理工大学

学习内容

- 3.1 输入输出系统概述
- 3.2 磁盘阵列
- 3.3 总线设计
- 3.4 通道处理机

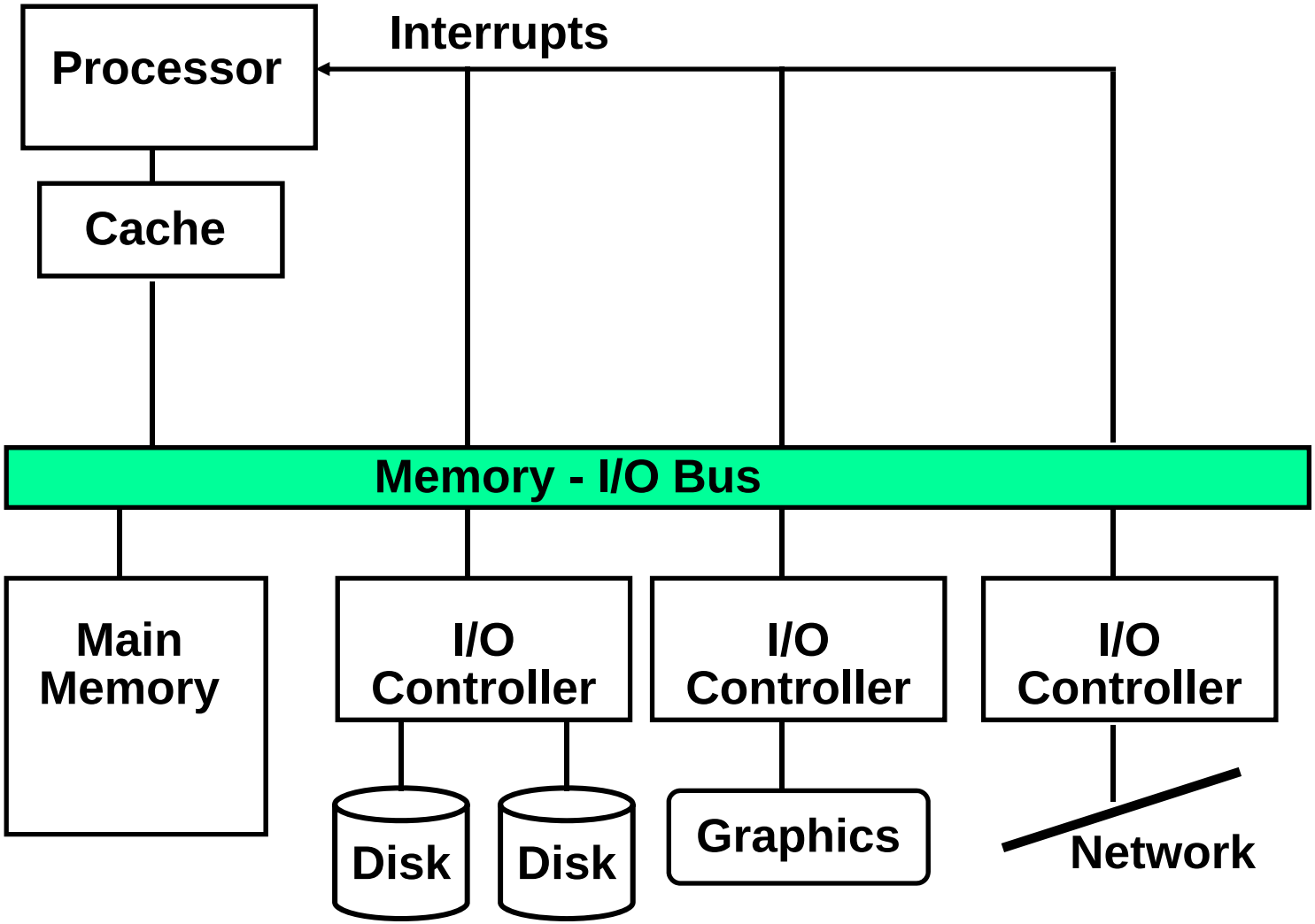
3.1 输入/输出系统概述

- 在计算机系统中，把处理机与主存储器之外的部分统称为输入输出系统（简称为I/O系统）。
- 输入输出系统是处理机与外界进行数据交换的通道。
- 输入输出系统包括：输入输出设备、设备控制器及与输入输出操作有关的软硬件。
- 与处理机有关的、除人以外的各种设备称为输入输出设备（或外围设备，I/O设备）。

3.1 输入/输出系统概述

- 输入输出系统是计算机系统中最具多样性和复杂性的部分。
- 最典型地反映了软硬件的相互结合。
- 目前，输入输出系统的复杂性都隐藏在操作系统中，需软、硬件人员共同解决。
- 设计目标：成本/性能/支持多种设备，同时避免I/O性能瓶颈。
  - 在主存和I/O设备之间进行平衡

3.1.1 输入/输出系统的组成与操作



3.1.1 输入/输出系统的组成与操作

- 组成
  - 设备控制器；
  - I/O设备
    - 磁盘、磁带、打印机、键盘、显示器等；
  - 与I/O操作有关的软件和硬件。

3.1.1 输入/输出系统的组成与操作

- 输入输出操作
  - 在低档单用户计算机上
    - 由程序员自己安排。
    - I/O系统设计主要考虑解决好CPU、主存和I/O设备之间巨大的速度差异。

3.1.1 输入/输出系统的组成与操作

- 输入输出操作（续）
  - 在大多数计算机上
    - 用户发出输入输出请求，由操作系统分配调度I/O设备，并进行具体的I/O处理，使I/O与CPU、主存的操作尽可能并行。
    - 在这样的系统上，I/O系统硬件的功能对应用程序员透明，I/O功能只反映在高级语言与操作系统的界面上。

3.1.1 输入/输出系统的组成与操作

- 因此，
  - I/O系统结构的设计应该是面向操作系统的；
  - 需要考虑如何在I/O系统与操作系统之间合理分配软、硬件功能。

3.1.2 输入/输出系统的功能与性能

- 输入输出系统的功能
  - 对指定外设进行I/O操，同时完成许多其它的管理和控制任务，如：
    - 编址
    - 准备通路
    - 信息传送
    - 格式变换
    - 中断请求

上述功能由三个部分协同完成：

- 输入输出机器指令；
- 输入输出设备及其控制器硬件；
- 操作系统；

3.1.2 输入/输出系统的功能与性能

- 随着输入输出数据量的增大，数据传送速度的明显增加和输入输出设备的日益增多，输入输出系统结构设计的好坏将直接影响计算机系统的性能。

3.1.2 输入/输出系统的功能与性能

- 输入输出系统的性能主要包括：
  - 输入输出速度；
  - 用户从输入到输出的等待时间；
  - CPU和主存的利用率；
  - I/O系统的兼容能力，可扩展能力，综合处理能力，性能价格比等。

3.1.2 输入/输出系统的功能与性能

- 误区：使用多进程技术可以忽略I/O性能对系统性能的影响。
- 多进程技术只能够提高系统吞吐率，并不能够减少系统响应时间；
- 进程切换时可能需要增加I/O操作；
- 可切换的进程数量有限，当I/O处理较慢时，仍然会导致CPU处于空闲状态。

3.1.2 输入/输出系统的功能与性能

- 例1：假设一台计算机的I/O处理占10%，当其CPU性能改进，而I/O性能保持不变时，系统总体性能会出现什么变化？
  - 如果CPU的性能提高10倍；
  - 如果CPU的性能提高100倍；

3.1.2 输入/输出系统的功能与性能

- 解：假设原来的程序执行时间为1个单位时间。
  - 如果CPU的性能提高10倍，程序的计算（包含I/O处理）时间为：

    $(1 - 10\%)/10 + 10\% = 0.19$

    即整机性能只能提高约 1/0.19 =5 倍，差不多有50％的CPU性能浪费在I/O上。

3.1.2 输入/输出系统的功能与性能

- 解：假设原来的程序执行时间为1个单位时间。
  - 如果CPU的性能提高100倍，程序的计算时间为：

    (1 - 10%)/100 + 10% = 0.109

    即整机性能只能提高约 1/0.109=10 倍，表示有90%的性能浪费在没有改进的I/O上了 。

3.1.2 输入/输出系统的功能与性能

- 例2：可以采取两种措施提高系统性能，问哪种措施更好，并给出理由。假设CPU有70%的时间用于处理，30%的时间用于等待磁盘访问。
  - 1. 升级CPU，处理速度提高50%。
  - 2. 采用新磁盘，其读写速度提高300%。

3.1.2 输入/输出系统的功能与性能

- 解：使用Amdahl定律。

1. Fe=0.7
   Se=1.5

$$S_{n(1)} = \frac{1}{(1-0.7)+\frac{0.7}{1.5}} = 1.30$$

2. Fe=0.3
   Se=4.0

$$S_{n(2)} = \frac{1}{(1-0.3)+\frac{0.3}{4.0}} = 1.29$$

哪一种？ **“Make the Common Case Faster”**

3.1.3 输入/输出系统的特点

- 异步性
  - 输入输出设备通常不使用统一的中央时钟，各个设备按照自己的时钟工作，但又要在某些时刻接受处理机的控制；
  - 处理机与外围设备之间，外围设备与外围设备之间能够并行工作。

3.1.3 输入/输出系统的特点

实时性

- 对于一般外部设备：可能丢失数据，或造成外围设备工作的错误；
- 对于实时控制计算机系统，如果处理机提供的服务不及时，可能造成巨大的损失，甚至造成人身伤害；
- 对于处理机本身的硬件或软件错误：如电源故障、数据校验错、页面失效、非法指令、地址越界等，处理机须及时处理；

对不同类型的设备，必须具有与设备相配合的多种工作方式。

3.1.3 输入/输出系统的特点

- 与设备无关性
  - 独立于具体设备的标准接口；
  - 例如：串行接口、并行接口、SCSI（Small Computer System Interface）接口等。

- 在需要更换外围设备时，各种不同型号，不同生产厂家的设备都可以直接通过标准接口与计算机系统连接。
- 处理机采用统一的硬件和软件对品种繁多的设备进行管理。
- 某些计算机系统已经实现了即插即用技术。

3.1.4 输入/输出系统的控制方式

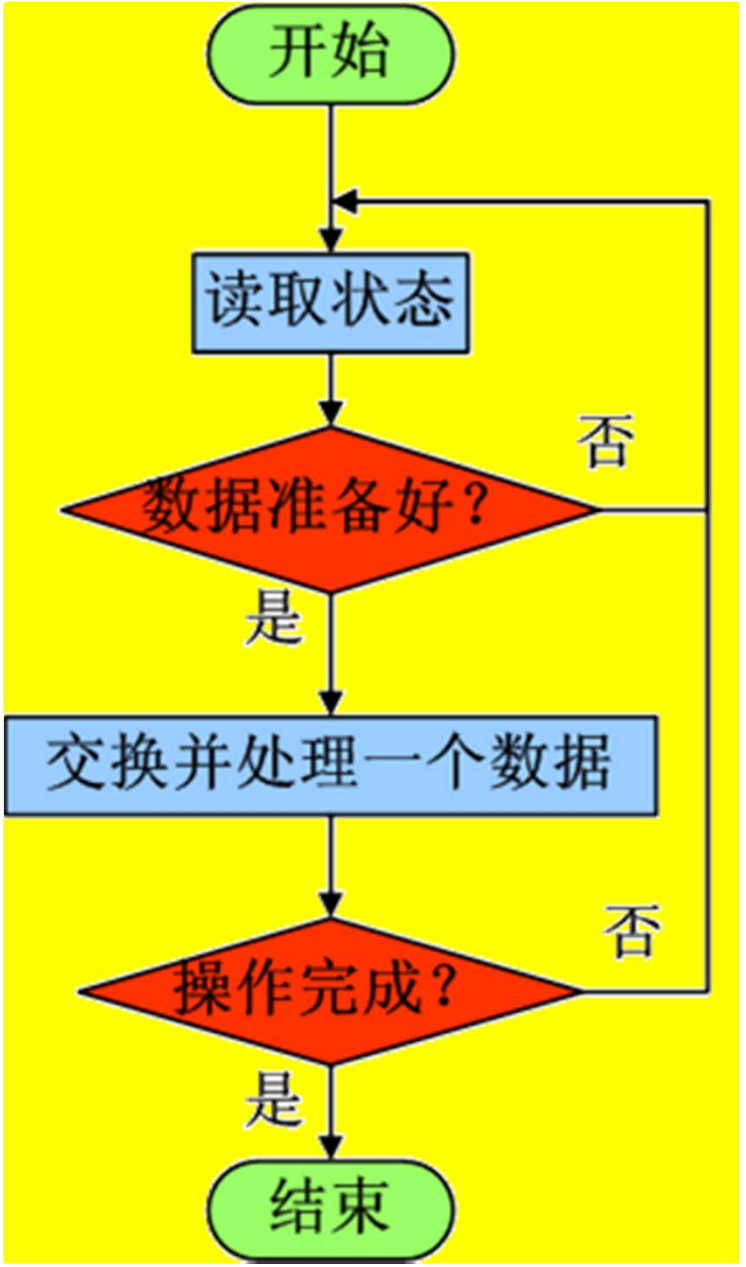
- 输入输出系统的发展经历了三个阶段，对应于三种控制方式：
  - 程序控制方式 与 中断方式
  - DMA(直接存贮器访问)方式
  - I/O处理机方式

程序控制方式

- **特点：CPU直接进行I/O操作。**
  - 何时、对何设备进行I/O操作完全受CPU控制；
  - CPU要通过指令对设备进行测试才能知道设备的工作状态。如空闲、准备就绪、正在忙碌等；
  - 数据的输入和输出都要经过CPU；
  - 用于连接低速外围设备，如终端、打印机等。

程序控制方式

- **工作原理：CPU查询外设已准备好后，才传送数据。**
- **特点：CPU与外设间通过程序同步，CPU被外设独占，CPU效率低下**
- **硬件要求：电路简单。**
- **应用： 适于在CPU不太忙且传送速度要求不高时。**



程序控制方式

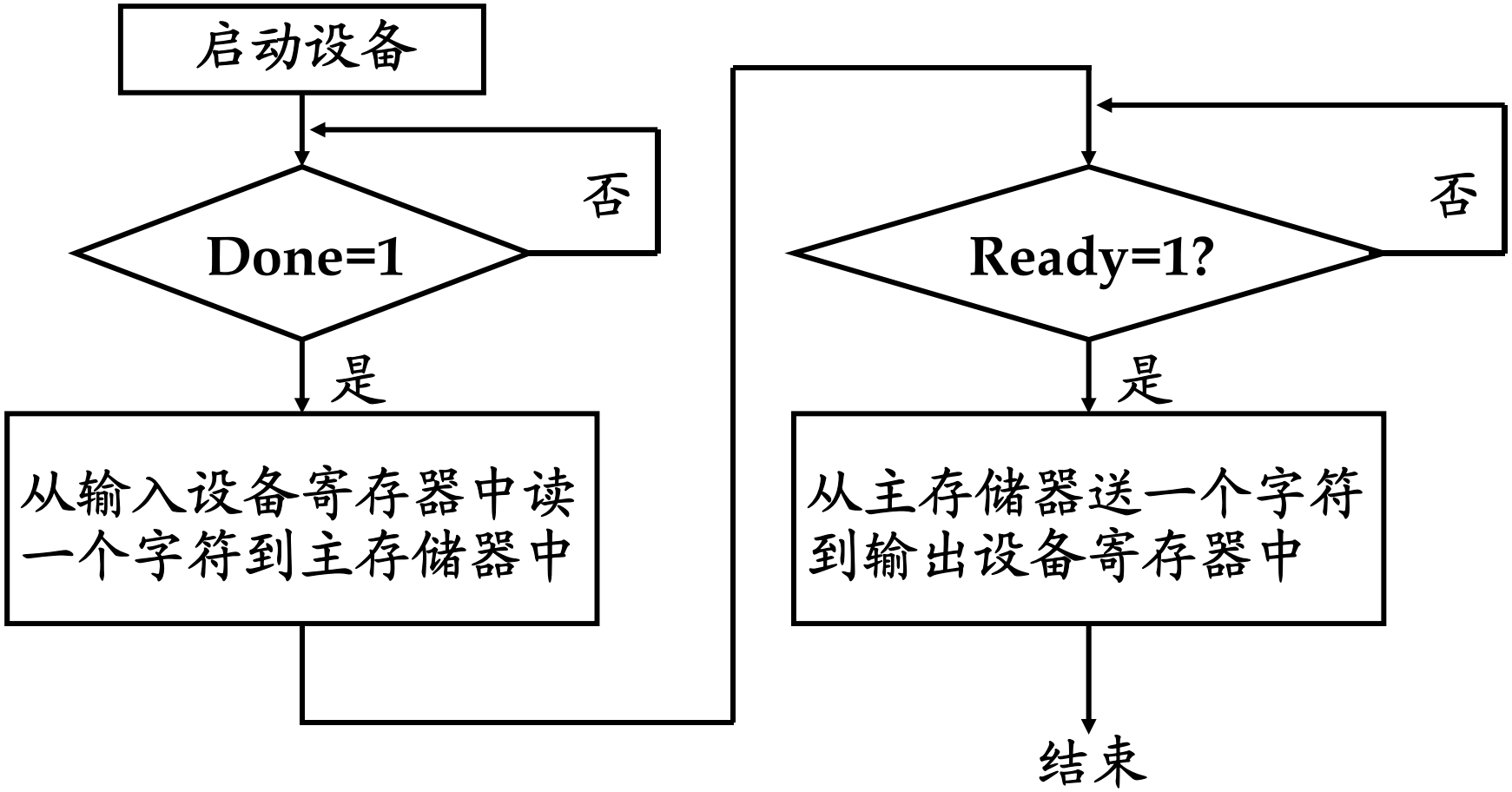


图 键盘输入再显示的框图

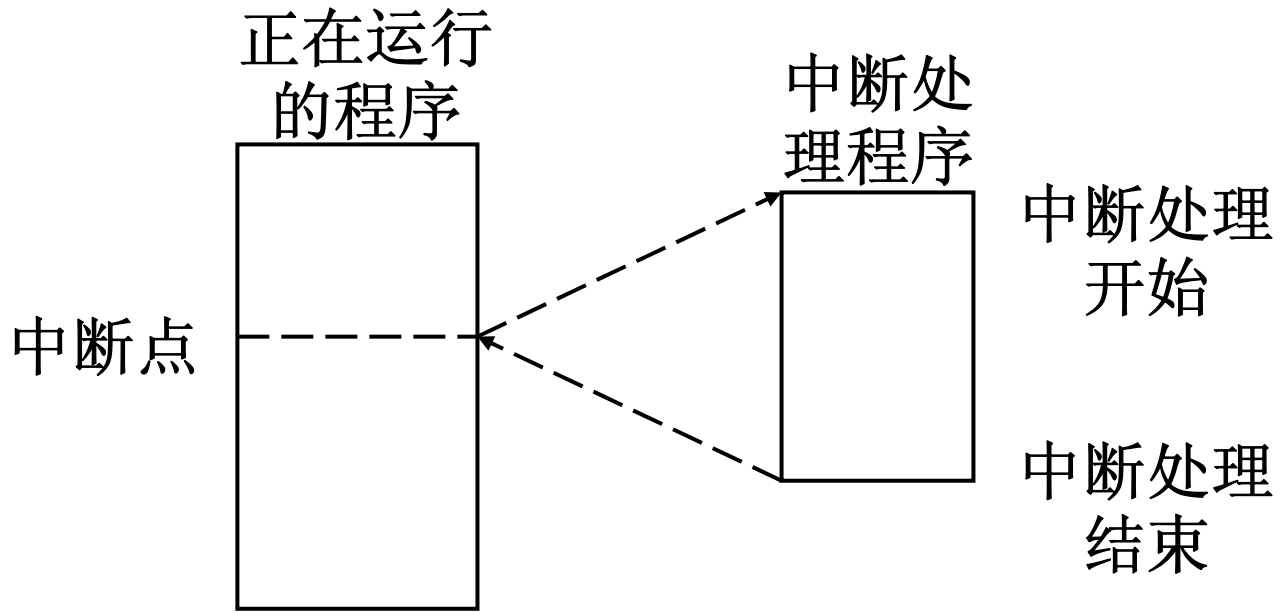
程序控制方式

- **优点：** 灵活性很好；可以很容易地改变各台外围设备的优先级。
- **缺点：** 浪费CPU资源，速度慢。实现处理机与外围设备并行工作困难。

中断方式

- 中断系统是I/O系统的重要组成部分。
- 中断源：引起中断的各种事件。
- 中断请求：中断源向中断系统发出请求中断的申请。
- 中断响应：允许CPU中断现行程序的运行，调用中断的处理程序。

中断方式



- **中断优先级：响应和处理中断的优先次序。**
- **输入输出中断是CPU与I/O设备及通道联系的工具，在输入输出请求/操作完成时或出现故障时发出。**

中断方式

- 数据的输入和输出都要经过CPU；
- 使CPU与外围设备能够并行工作；
- 能够处理例外事件。例如，电源掉电等；
- 灵活性好；
- 用于连接低速外围设备。

DMA方式

- DMA：直接存贮器访问
- 特点
  - 在主存与外设之间有直接访问通路；
  - CPU挪用一个存储周期启动DMA操作；
  - 在DMA方式开始和结束时，需要处理机进行管理；
  - 在DMA方式的数据传送过程不需要CPU干预。

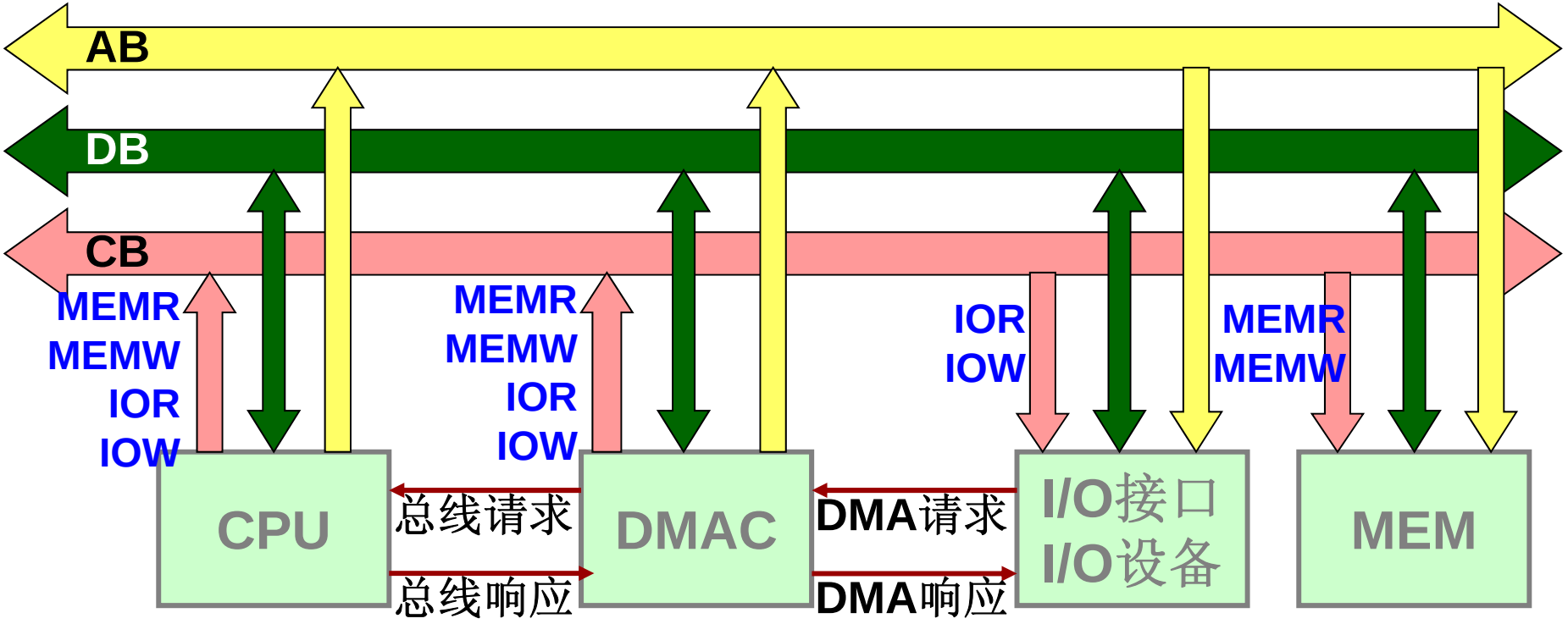
DMA方式

- 优点：输入输出与CPU并行工作，提高了速度和效率
- 缺点：外设的管理、DAM启动、数据准备、操作完毕后的处理还需由CPU完成
- 主要用来连接高速外围设备，例如磁盘

DMA过程

- ①预处理：CPU初始化DMAC（设置存储器和外设的起始地址、传送字节数），然后启动DMAC工作。
- ②数据传送：DMAC控制存储器和IO设备之间的数据交换。
- ③后处理：传送完成后， DMAC向CPU发中断请求，CPU收尾处理。
- CPU在执行完当前指令的当前总线周期后，就响应DMA请求。

DMA方式控制的数据传送



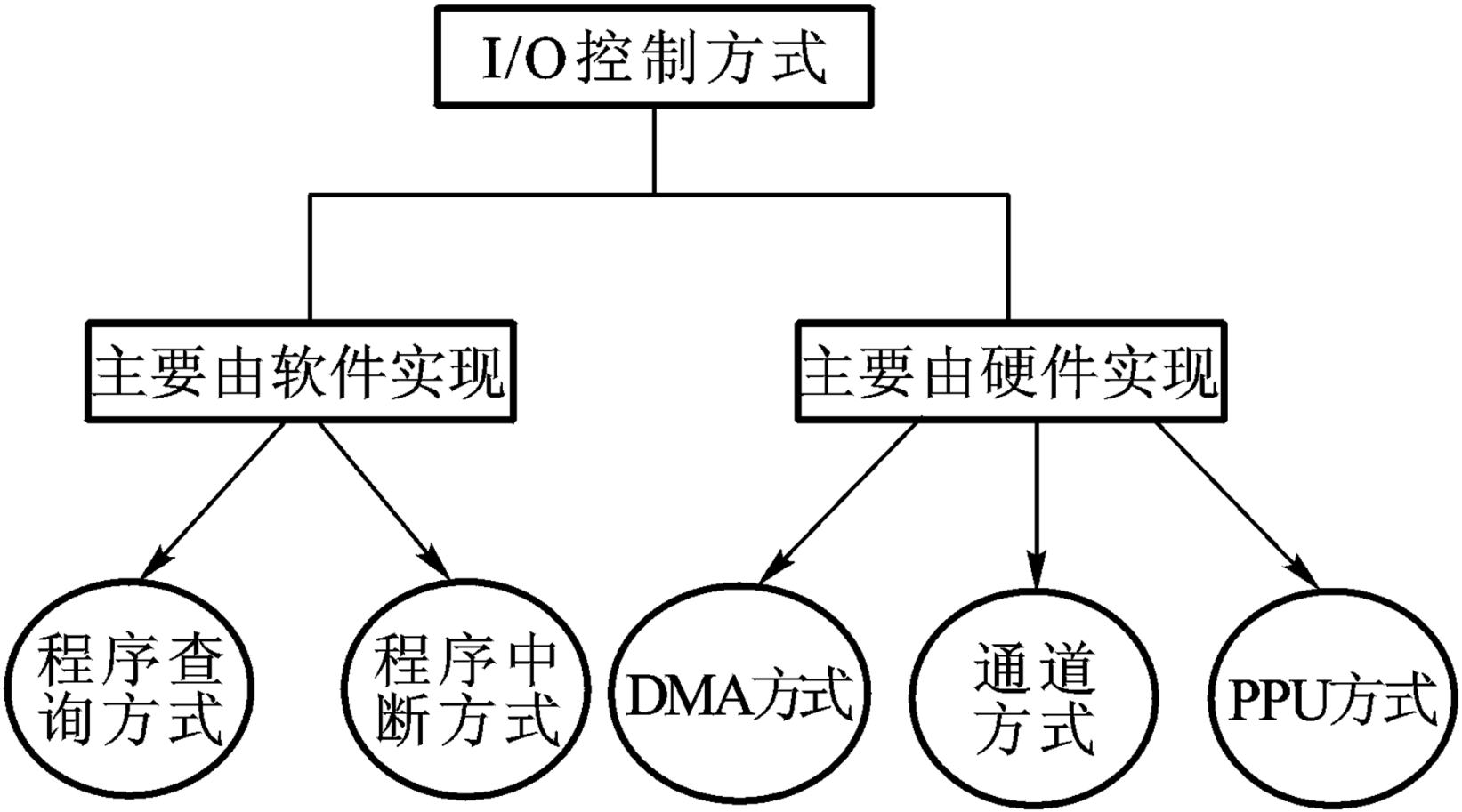
DMA传送方向：

- （1）存储单元传送：存储器→存储器。
- （2）DMA读传送：存储器→I/O设备。
- （3）DMA写传送：I/O设备→存储器。

I/O处理机方式

- 特点：
  - 把对外围设备的输入输出和管理工作从CPU分离出来，由专门的处理机完成；
  - 实现CPU与I/O设备操作的并行。
- 两种形式：
  - 通道（或通道处理机）方式
  - 外围处理机方式

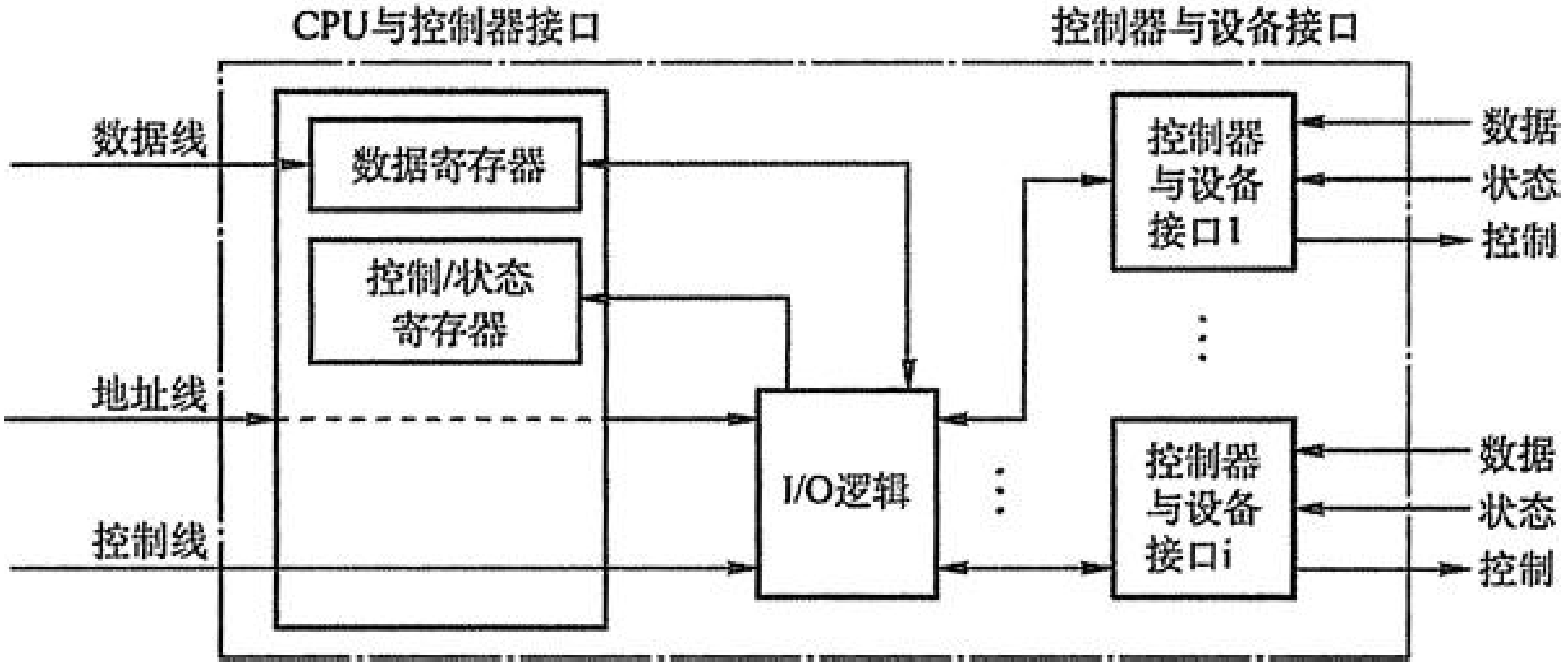
3.1.4 输入/输出系统的控制方式



3.1.5 输入/输出系统的组织

- 针对异步性，采用自治控制的方法。
- 针对实时性，采用层次结构的方法。
- 针对与设备无关性，采用分类处理方法。

3.1.5 输入/输出系统的组织



自治控制

- 输入输出系统是一个独立于处理机之外的自治系统。
- 处理机与外围设备之间要有恰当的分工。

层次结构

- 最靠近处理机的是输入输出处理机、输入输出通道等。
- 中间层是标准接口。标准接口通过设备控制器与输入输出设备相连接。
- 设备控制器控制外围设备工作。

分类处理

- 面向字符的设备：
  - 指工作速度比较低的机电类设备。
  - 例如，字符终端、打字机等。
- 面向数据块的设备：
  - 主要指工作速度比较高的外围设备；
  - 例如，磁盘、磁带、光盘的辅助存储器，行式打印机等。

分类处理

Device	Behavior	Partner	Data rate (Mb/s)
Keyboard	input	human	0.0001
Mouse	input	human	0.0038
Laser printer	output	human	3.2000
Graphics display	output	human	800.0000-8000.0000
Network/LAN	input or output	machine	100.0000-1000.0000
Magnetic disk	storage	machine	240.0000-2560.0000

3.2 磁盘阵列（RAID）

- Redundant Array of Inexpensive Disks，廉价冗余磁盘阵列。
- Redundant Array of Independent Disks，独立冗余磁盘阵列。
- 将一组磁盘驱动器用某种逻辑方式联系起来，作为逻辑上的一个磁盘驱动器来使用。

Disk：60 Years Old!

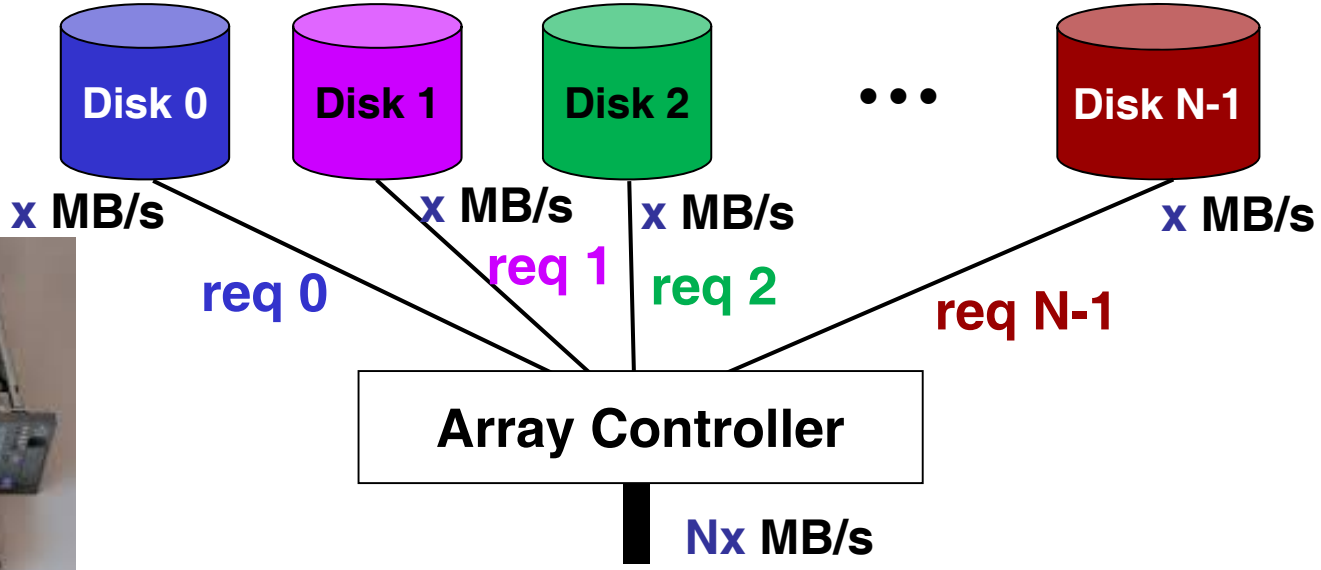
- 13th September 1956
- The IBM RAMAC 350

Disk：60 Years Old!

- 80000 times more data on the 8GB 1-inch drive in his right hand than on the 24-inch RAMAC one in his left…

RAID : 40 Years Old!

- Invented by David Patterson, Garth A. Gibson, and Randy Katz at the University of California, Berkeley in 1987



RAID优点

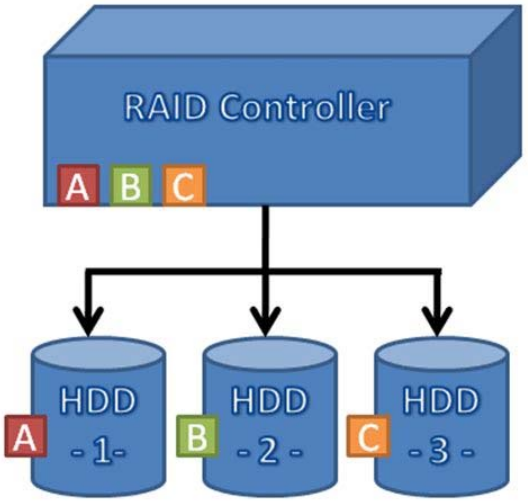
- ① 成本低，功耗小，传输速率高。
  - 很多磁盘驱动器同时传输数据， RAID可以达到单个磁盘驱动器几倍、几十倍甚至上百倍的速率。
- ② 提供容错功能，安全性更高。
- ③ 同样容量下，价格要低许多。

RAID的分级

RAID级别	名称	最少数据磁盘数	可正常工作的最多失效磁盘数	检测磁盘数
RAID 0	无冗余无校验的磁盘阵列	2	0	0
RAID 1	镜象磁盘阵列	2	1	0
RAID 2	纠错海明码磁盘阵列	2	1	1
RAID 3	位交叉奇偶校验的磁盘阵列	2	1	1
RAID 4	块交叉奇偶校验的磁盘阵列	2	1	1
RAID 5	无独立校验盘的奇偶校验磁盘阵列	2	1	1
RAID 6	双维无独立校验盘的奇偶校验磁盘阵列	2	2	2

磁盘阵列（RAID 0）

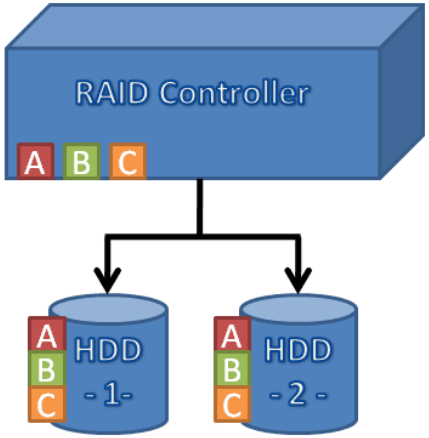
- 亦称数据分块（Striping），即把数据分布在多个盘上，实际上是非冗余阵列，无冗余信息。严格地说，它不属于RAID系列。

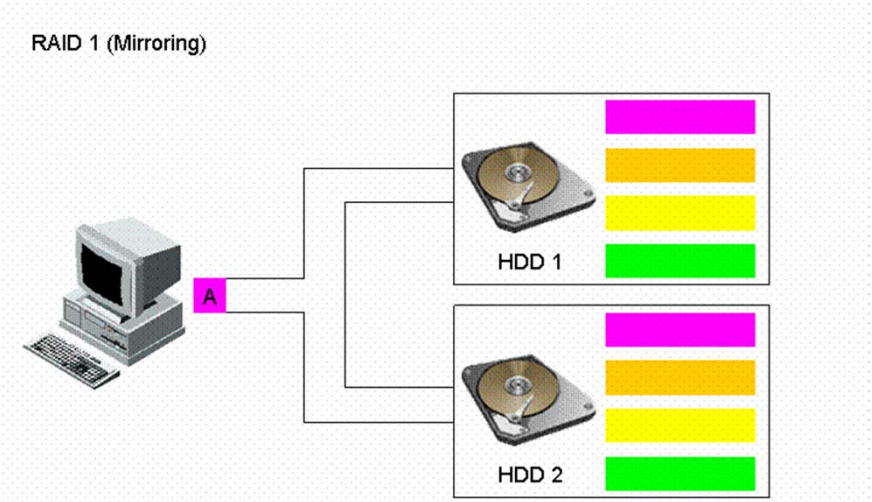


  - 优点：高性能，磁盘利用率高。
  - 缺点：系统可靠性差。

磁盘阵列（RAID 1）

- 亦称磁盘镜像，使用双备份磁盘。
- 需要一对硬盘

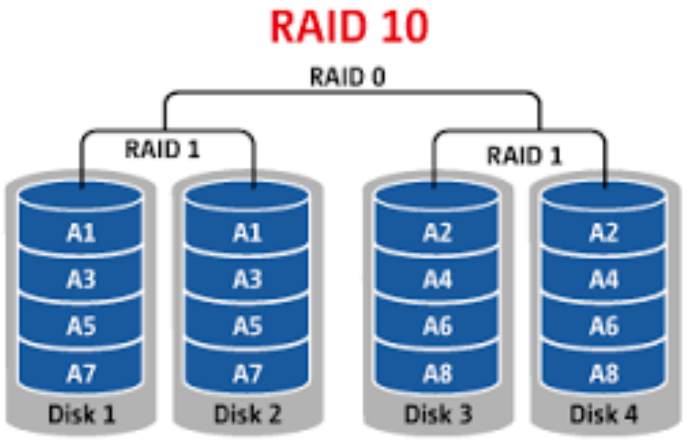


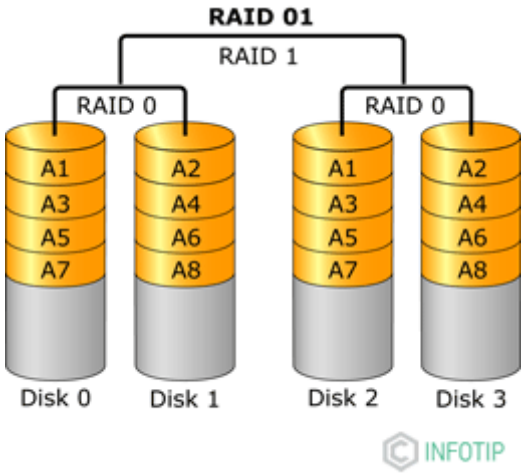


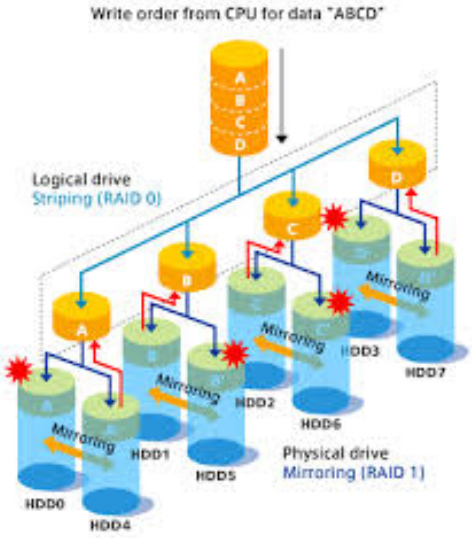
  - 优点：系统可靠性好，设计简单。
  - 缺点：硬件开销大；效率低。
  - 应用领域：高可靠性数据存储领域，如金融等。

磁盘阵列（RAID 0+1）

- 也被称为RAID 10 标准
- RAID 0和RAID 1标准结合的产物



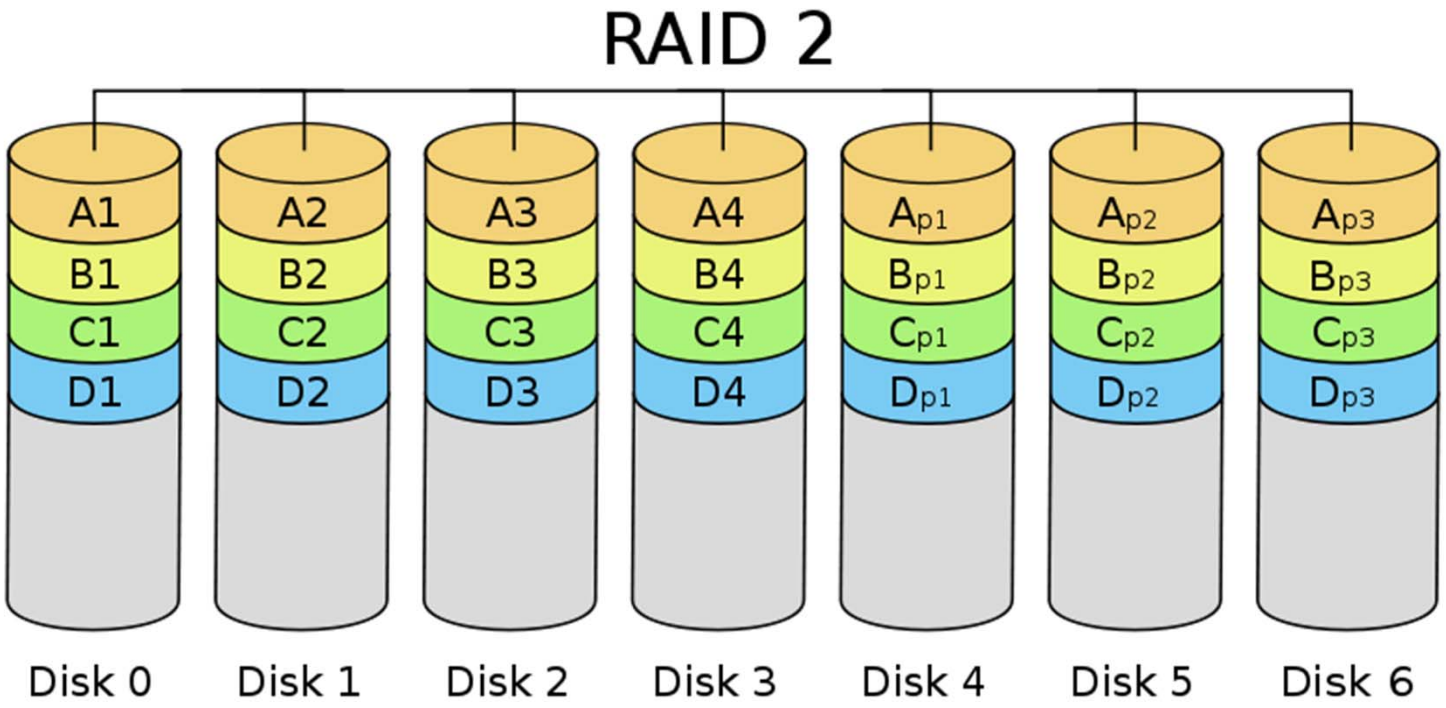




- 优点：同时拥有RAID 0的速度和RAID 1的数据高可靠性。
- 缺点：CPU占用率同样也更高，磁盘的利用率比较低。

磁盘阵列（RAID 2）

- 位交叉式海明编码阵列。
- 需要至少三块硬盘。

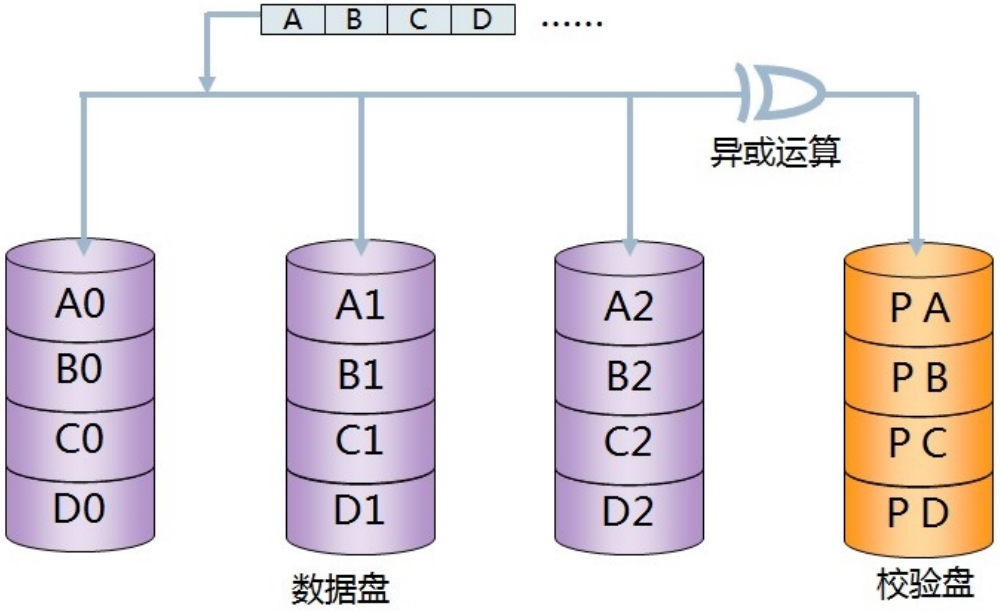


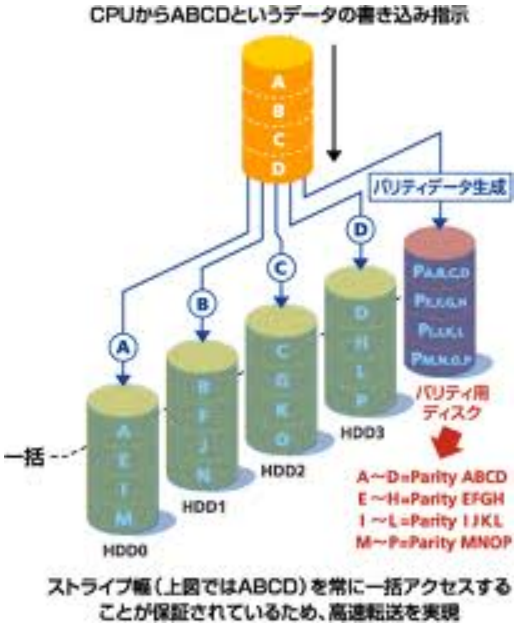
磁盘阵列（RAID 2）

- 优点：
  - 高速误差校正；
  - 数据传输速率高；
  - 和后面各级相比，设计较简单。
- 缺点：
  - 校正空间较大，盘阵列利用率较低；
  - 需要同步控制。

磁盘阵列（RAID 3）

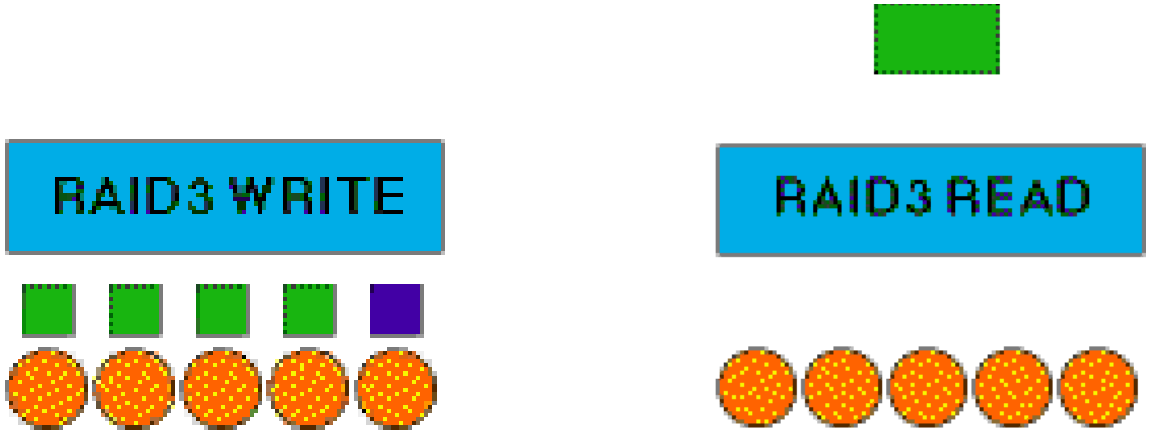
- 位交叉奇偶校验盘阵列，是单盘容错并行传输的阵列。即数据以位或字节交叉的方式存于各盘，冗余的奇偶校验信息存储在一台专用盘上。
- 需要至少三块硬盘。





磁盘阵列（RAID 3）

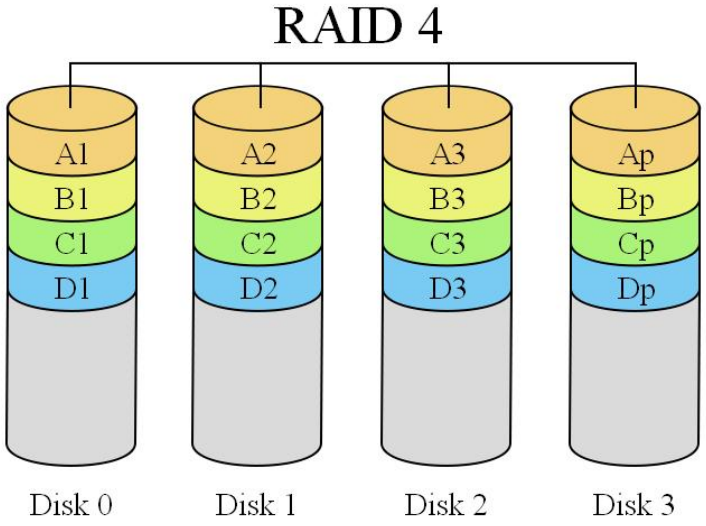
- 在RAID 3中，将磁盘分组，读写要访问组中所有盘。当一个磁盘出故障时，可以通过奇偶校验磁盘中的校验和来恢复出错数据。



  - 优点：读写速度快；磁盘失效对吞吐率的影响较小；盘阵列利用率高。
  - 应用领域：多媒体应用。

磁盘阵列（RAID 4）

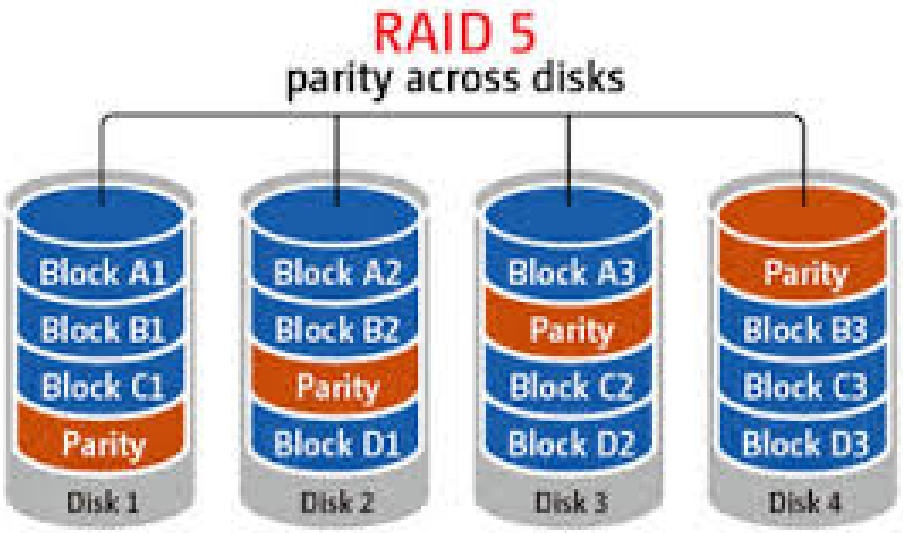
- 专用奇偶校验独立存取盘阵列。即数据以块(块大小可变)交叉的方式存于各盘，冗余的奇偶校验信息存在一台专用盘上。



- 优点：读写速度快；盘阵列利用率高；
- 缺点：设计较复杂，磁盘失效对吞吐率的影响较大。

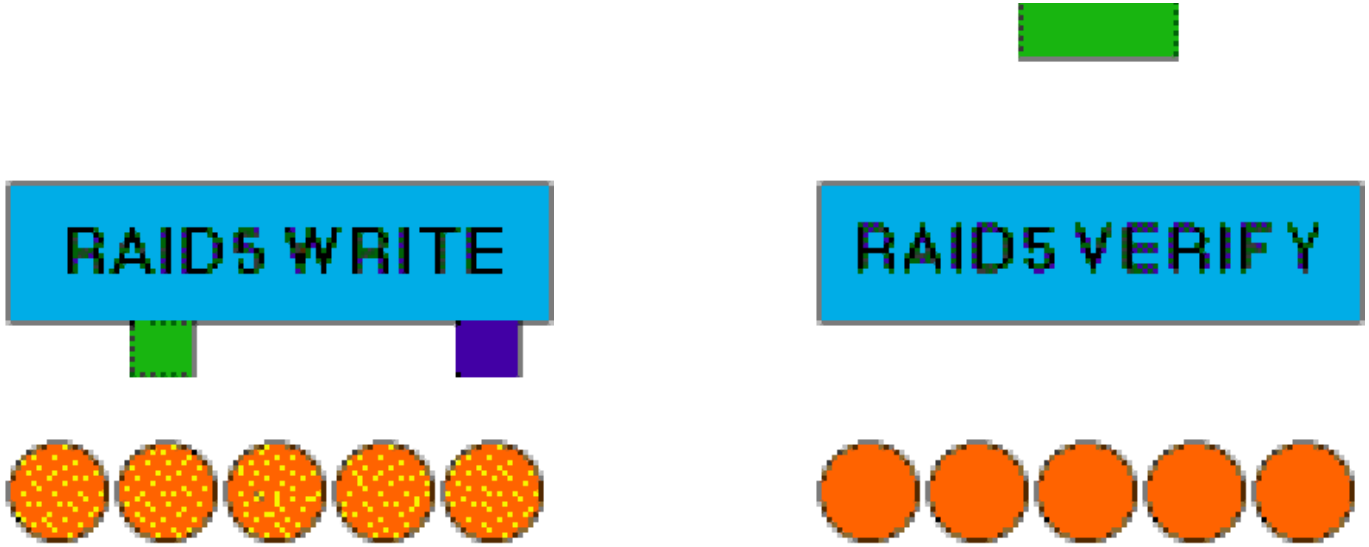
磁盘阵列（RAID 5）

- 块交叉分布式奇偶校验盘阵列，即数据以块交叉的方式存于各盘，但无专用的校验盘，而是把冗余的奇偶校验信息均匀地分布在所有磁盘上。至少需要三块磁盘。



磁盘阵列（RAID 5）

- 对于RAID 5，数据块每一行的相联奇偶校验不再限制在单一磁盘上，只要块单元不位于同一个磁盘内，这种组织方法就可以支持多个写同时执行。可容忍单盘出错。

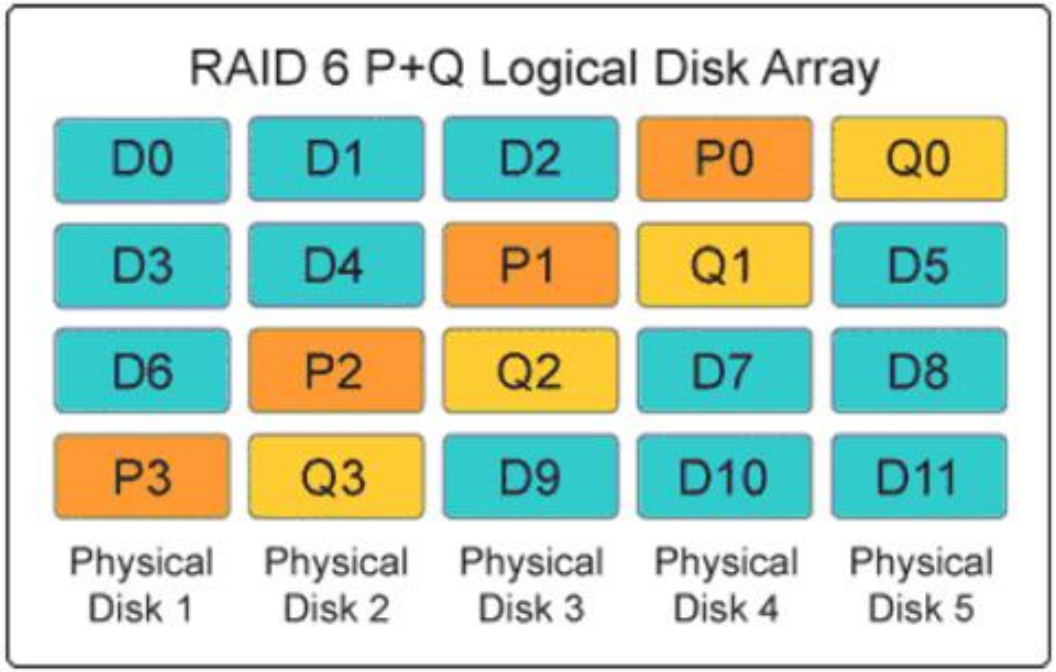


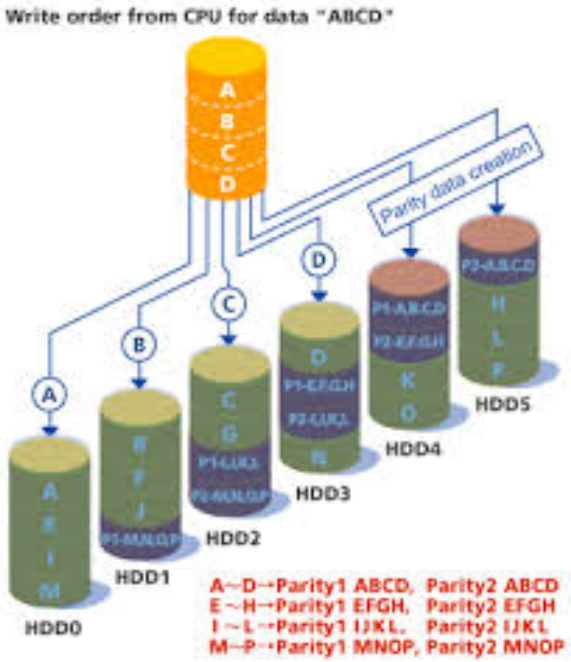
磁盘阵列（RAID 5）

- 优点：
  - 读数据速率最高；
  - 盘阵列利用率高。
- 缺点：
  - 磁盘失效对系统可靠性有一定影响；
  - 控制器设计复杂。
- 应用领域：
  - 文件、数据、Web服务器等。

磁盘阵列（RAID 6）

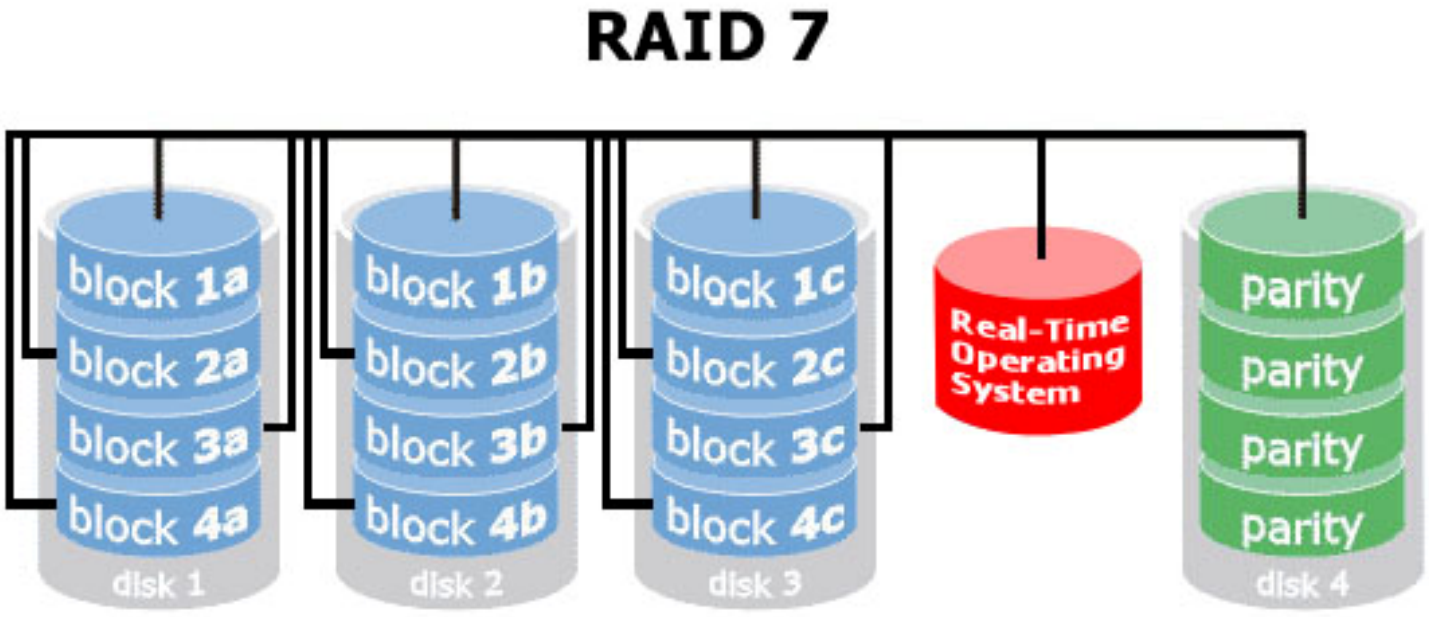
- 双维奇偶校验独立存取盘阵列。即数据以块(块大小可变)交叉的方式存于各盘，冗余的检、纠错信息均匀地分布在所有磁盘上。并且，每次写入数据都要访问一个数据盘和两个校验盘，可容忍双盘出错。





磁盘阵列（RAID 7）

- RAID 7是采用Cache和异步技术的RAID 6，使响应速度和传输速率有了较大提高。



3.2 磁盘阵列（RAID）

级别	技术	描述	速度	容错能力
RAID 0	磁盘分段	没有校验数据	磁盘并行I/O，存取速度提高最大	数据无备份
RAID 1	磁盘镜像	没有校验数据	读数据速度有提高	数据100％备份
RAID 2	磁盘分段＋海明码数据纠错	专用校验数据盘	没有提高	允许单个磁盘错
RAID 3	磁盘分段＋奇偶校验	专用校验数据盘	磁盘并行I/O，速度提高较大	允许单个磁盘错，校验盘除外。
RAID 4	磁盘分段＋奇偶校验	异步专用校验数据盘	磁盘并行I/O，速度提高较大	允许单个磁盘错，校验盘除外。
RAID 5	磁盘分段＋奇偶校验	校验数据分布存放于多盘	磁盘并行I/O，速度提高较大，比RAID 0稍慢	允许单个磁盘错，无论哪个盘。
RAID 6	磁盘分段＋双校验	校验数据分布存放于多盘	RAID5 扩展	允许两个磁盘错，无论哪个盘。

3.3 总线设计

- 在大多数小型和微型计算机系统中，计算机的各子系统之间通过**总线（Bus）**实现连接。

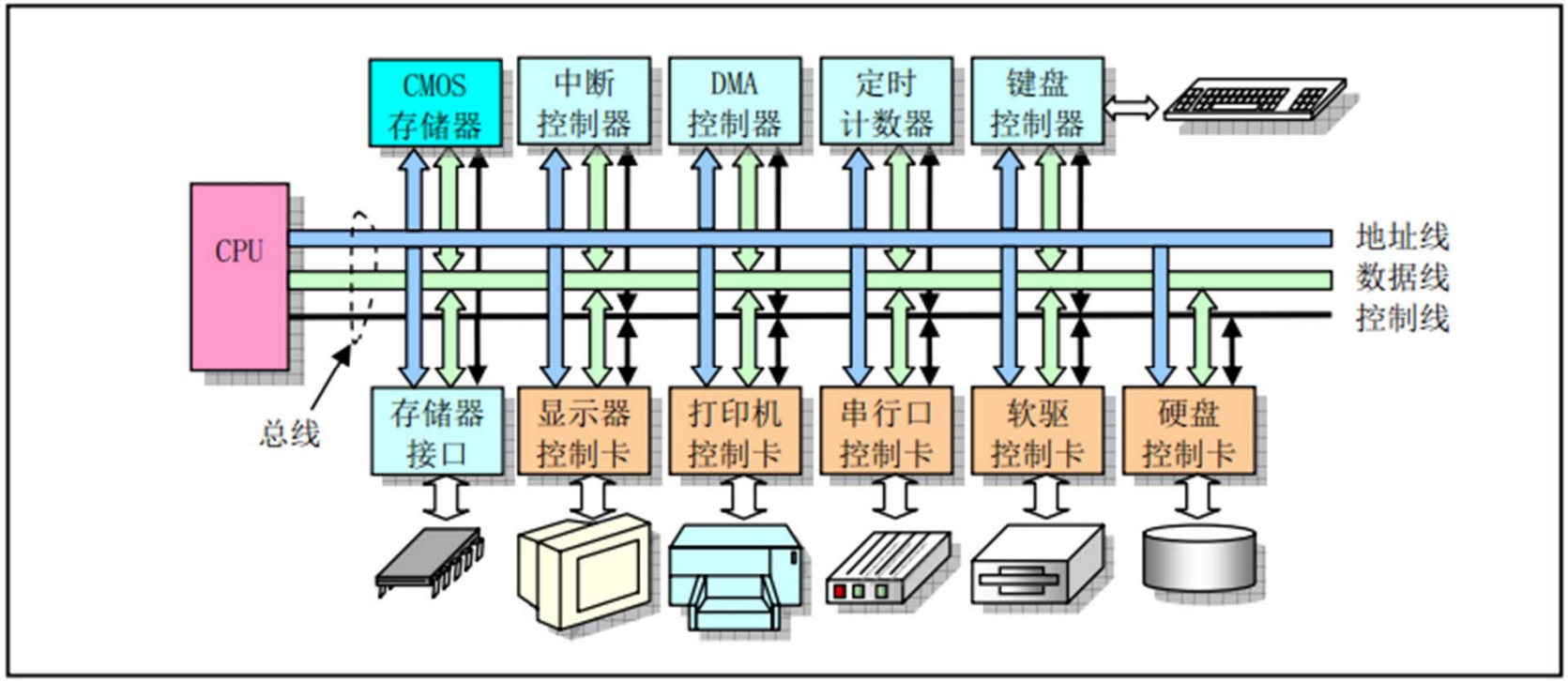


图 2-2 传统 IBM PC 及其兼容计算机的组成框图

3.3 总线设计

- 总线设计存在很多技术难点，因为：
  - 总线上信息传送的速度极大地受限于各种物理因素。
    - 如总线的长度、设备的数目、信号的强度等，这些物理因素限制了总线性能的提高。
  - 另外，我们一方面要求I/O操作响应快，另一方面又要求高吞吐量，这可能造成设计需求上的冲突。

3.3 总线设计

设计总线时需要考虑的一些问题

特性	高性能	低价格
总线宽度	独立的地址和数据总线	数据和地址分时 共用同一套总线
数据总线宽度	越宽越快（例如：64位）	越窄越便宜（例如：8位）
传输块大小	块越大总线开销越小	单字传送更简单
总线主设备	多个（需要仲裁）	单个（无需仲裁）
分离事务	分离的请求包和回答包能提高总线带宽	不采用。持续连接成本更低，而且延迟小
定时方式	同步	异步

3.3.1 总线特点

- 总线是一组能为多个部件分时共享的公共信息传送线路。
  - 共享是指总线上可以挂接多个部件，各个部件之间相互交换的信息都可以通过这组公共线路传送；
  - 分时是指同一时刻总线上只能传送一个部件发送的信息。

3.3.1 总线特点

- 优点：
  - 成本低，简单；
  - 易用：若遵循总线标准，很容易增加或移动设备。
- 缺点：
  - 总线的带宽形成了信息交换的瓶颈，从而限制了系统中总的I/O吞吐量。

总线事务

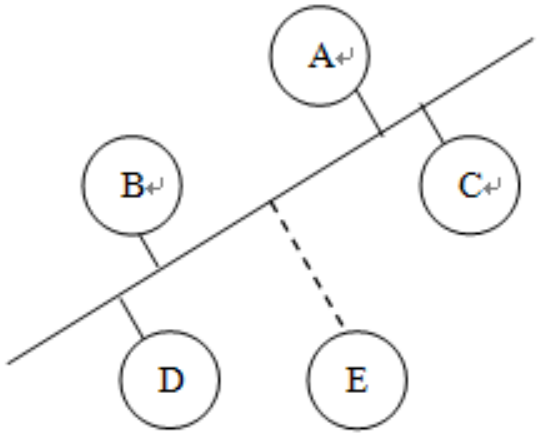
- 通常把在总线上一对设备之间的一次信息交换过程称为一个“总线事务”。
- 把发出总线事务请求的部件称为主设备，与主设备进行信息交换的对象称为从设备。
  - 例如CPU要求读取存储器中某单元的数据，则CPU是主设备，而存储器是从设备。
- 总线事务类型通常根据它的操作性质来定义，典型的总线事务类型有“存储器读”、“存储器写”、“I/O读”、“I/O写”、“中断响应”等，
- 一次总线事务简单来说包括两个阶段：地址阶段和数据阶段。

总线使用权

- 主设备发出总线请求并获得总线使用权后，就立即开始向从设备进行一次信息传送，称为主从关系。
- 主设备负责控制和支配总线，向从设备发出命令来指定数据传送方式与数据传送地址信息。只有获得总线使用权的设备才是主设备。
- 通常，将完成一次总线操作的时间称为总线周期。总线使用权的转让发生在总线进行一次数据传送的结束时刻。

总线类型

- 多种分类方法
- 按信息传送的方向：
  - 单向传输总线
  - 双向传输总线
- 按用法：
  - 专用的
  - 非专用的



采用非专用总线连接各个部件，增设部件很方便

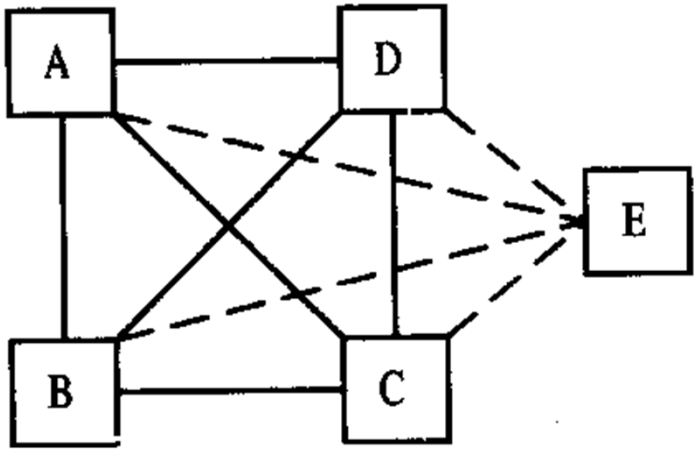


图 所有部件之间用专用总线互连

总线类型

- **片内总线和片外总线：**
  - **片内总线是CPU内部的寄存器、算术逻辑部件、控制部件以及总线接口部件之间的公共信息通道。**
  - **片外总线则泛指CPU与外部器件之间的公共信息通道。**
  - **有的资料上也把片内总线叫做内部总线或内总线(Internal Bus)，把片外总线叫做外部总线或外总线(External Bus)。**

总线类型

- **CPU总线、系统总线和外设总线**
  - **CPU总线**是从CPU引脚上引出的连接线，用来实现CPU与外围控制芯片和功能部件之间的连接。
  - **系统总线**也称为I/O通道总线，用来与存储器和扩充插槽上的各扩充板卡相连接。常见的系统总线有ISA、PCI、PCI-E等。系统总线是通过专用的逻辑电路的对CPU总线的信号在空间与时间上进行逻辑重组转换而来。
  - **外设总线**是指计算机主机与外部设备接口的总线，实际上是一种外设的接口标准。目前在微型计算机上流行的接口标准有：IDE（EIDE/ATA，SATA）、SCSI、USB和IEEE 1394四种。前两种主要用于连接硬盘、光驱等外部存储设备，后面两种可以用来连接多种外部设备。

3.3.2 总线性能指标

（1）总线宽度（1/2）

- 总线宽度指的是总线的线数，它决定了总线所占的物理空间和成本。
- 对总线宽度最直接的影响是地址线和数据线的数量。主存空间和I/O空间的扩充使地址线数量的增加，并行传输要求有足够的数据线。如64位数据线和64位地址线在高档微机中已较为普遍，在大型高性能计算机中数据线和地址线更多。

3.3.2 总线性能指标

（1）总线宽度（2/2）

- 在满足性能要求以及所用通信类型和速率适配的情况下，应尽量减少总线的线数。
- 通过采用线的组合、并/串–串/并转换和编码可以减少总线的线数，但这通常会降低总线的流量。

3.3.2 总线性能指标

- 例1：使用ISA总线（20位地址线）允许寻址的主存空间有多大？使用PCI总线（32位地址线）允许寻址的主存空间又有多大？
- 解：

ISA总线的主存空间＝2^{20}个主存单元＝1M个主存单元

PCI总线的主存空间＝2^{32}个主存单元＝4G个主存单元

3.3.2 总线性能指标

- （2）总线带宽
  - 总线带宽是指总线的最大数据传输速率，即每秒传输的字节数。
  - 在同步通信中，总线的带宽与总线时钟密不可分，总线时钟频率的高低决定了总线带宽的大小。总线带宽＝总线宽度×总线频率
- 总线的实际带宽还会受到总线长度（总线延迟）、总线负载、总线收发器性能等多方面因素的影响。

3.3.2 总线性能指标

- 例2：PCI总线的时钟频率为33MHz/66MHz，当该总线进行32/64位数据传送时，总线带宽各是多少？
- 解：假设一个总线时钟周期*T*完成一个数据的传送，时钟频率为*f*，数据位为*n*，总线带宽用*Dr*表示，则

$$Dr = \frac{n}{8 \times T} = \frac{n \times f}{8}$$

假设$f = 33\text{MHz} = 33 \times 10^6/\text{s}$，$n = 32$位，根据定义可得

$$Dr = (32/8) \times 33 \times 10^6/\text{s} = 132\text{MB/s}$$

3.3.2 总线性能指标

- 例3：假设某系统总线在一个总线周期中并行传输4字节信息，一个总线周期占用2个时钟周期，总线时钟频率为10MHz，求总线带宽。
- 解：因为一个总线周期占用2个时钟周期，完成一个32位数据的传送。总线时钟频率 $f=10\text{MHz}$，时钟周期 $T=1/f=0.1\mu\text{s}$，总线周期 $=2T=0.2\mu\text{s}$。一个总线周期中并行传输4字节信息，则总线带宽是 $4\div0.2=20\text{MB/s}$。

3.3.2 总线性能指标

（3）总线负载

- 总线负载是指连接在总线上的最大设备数量。大多数总线的负载能力是有限的。

（4）总线复用

- 总线分时复用是指在不同时段利用总线上同一个信号线传送不同信号，例如地址总线和数据总线共用一组信号线。采用这种方式的目的是减少总线数量，提高总线的利用率。

3.3.2 总线性能指标

（5）总线猝发传输

- 猝发式（突发式）数据传输是一种总线传输方式，即在一个总线周期中可以传输存储地址连续的多个数据。

总线的数据宽度与总线线数

- 数据宽度是I/O设备取得I/O总线后所传送数据的总量。
  - 数据宽度有单字（单字节）、定长块、变长块、单字加定长块和单字加变长块等。
- 数据通路宽度是数据总线的物理宽度，也就是数据总线的线数（数据总线宽度）。
- 两次分配总线期间所传送的数据宽度可能要经过多个时钟周期分次传送才能完成。

总线的数据宽度与总线线数

- 单字（单字节）宽度适合于低速设备。因为这些设备在每次传送一个字（字节）后的访问等待时间很长，在这段时间里让总线释放出来为别的设备服务，可大大提高总线利用率和系统效率。
- 定长块宽度适合于高速设备，可以充分利用总线带宽。定长块也不用指明传送信息的长度，简化了控制。但由于块的大小固定，当它要比实际传送的信息块小得多时，仍要多次分配总线。

总线的数据宽度与总线线数

- 变长块宽度适合于高优先级的中高速设备，灵活性好，可按设备的特点动态地改变传送块的大小，使之与部件的物理或逻辑信息块的大小一致。
- 单字加定长块宽度适合于速度较低而优先级较高的设备。这样，定长块的大小就不必选择过大，信息块超过定长块的部分可用单字处理，从而减少总线带宽、部件的缓冲器空间，减少部件可用能力的浪费。

总线传输技术问题

- 由于总线是多个设备共用的，所以系统中的设备使用总线必须要受到控制。
- 设备使用总线的步骤是：

  总线请求、总线仲裁、寻址、传送数据、检错和报错。

- 总线控制线路包括：

  总线仲裁逻辑、驱动器和中断逻辑等。

总线传输技术问题

- 总线传输中需要解决的问题主要包括：
  - 总线传输同步。为使信息正确传送，防止丢失，需对总线通信进行定时。
  - 总线仲裁控制。在总线上某一时刻只能有一个总线主部件控制总线，为避免多个部件同时发送信息到总线的矛盾，需要有总线仲裁机构。
  - 出错处理。数据传送过程中可能产生错误，有些部件有自动纠错电路，可以自动纠正错误。有些部件虽无自动纠错能力，但能发现错误，这时可发出“数据出错”信号，通知CPU来进行处理。
  - 总线驱动。通常采用三态输出电路或集电极开路输出电路来驱动总线。

3.3.3 总线定时控制

- 两种方式：同步 和 异步
- 同步定时方式
  - 系统采用一个统一的时钟信号来协调发送和接收双方的传送定时关系。时钟产生相等的时间间隔，每个间隔构成一个总线周期。在一个总线周期中，发送和接收双方可以进行一次数据传送。
  - 同步方式中的时钟频率必须能适应在总线上最长的延迟和最慢的接口的需要。

3.3.3 总线定时控制

异步定时方式

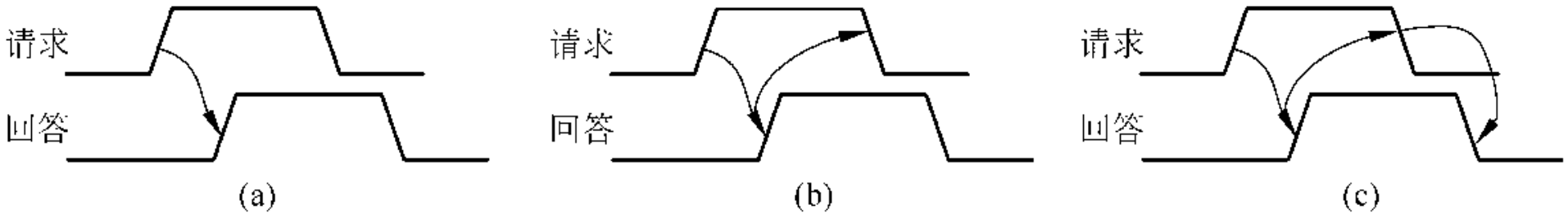
- 也称为应答方式。
- 在这种方式下，没有公用的时钟，也没有固定的时间间隔，完全依靠传送双方相互制约的“握手”信号来实现定时控制。
- 异步控制能保证两个工作速度相差很大的部件或设备间可靠地进行信息交换，但控制较同步方式稍复杂一些，成本也会高一些。

异步定时方式互锁

- 异步方式根据“请求”和“回答”信号的撤销是否互锁，有3种情况（1/2）：
- （1）不互锁(图a)
  - “请求”和“回答”信号都有一定的时间宽度，“请求”信号的结束和“回答”信号的结束不互锁。
- （2）半互锁(图b)
  - “请求”信号的撤销取决于接收到“回答”信号，而“回答”的撤销由从设备自己决定。

异步定时方式互锁

- 异步方式根据“请求”和“回答”信号的撤销是否互锁，有3种情况（2/2）：
- （3）全互锁(图c)
  - “请求”信号的撤销取决于“回答”信号的来到，而“请求”信号的撤销又导致“回答”信号的撤销。最高的灵活性和可靠性，付出了接口电路复杂性的代价



总线通信技术

1. 同步通信

- 两个部件之间的信息传送是通过定宽、定距的系统时标进行同步的。
- 这种方式的信息传送速率高，受总线的长度影响小，但会因时钟在总线上的时滞而造成同步误差，且时钟线上的干扰信号易引起误同步。

总线通信技术

2. 异步通信

- 由于I/O总线一般是为具有不同速度的许多I/O设备所共享，因此宜于采用异步通信。
- 异步通信又可分为单向控制和双向(请求/回答)控制两种。
- 单向控制指的是通信过程只由目的或源部件中的一个控制。单向控制又有源控制和目的控制两种。
- 双向控制是由源和目的双方共同控制。

总线通信技术

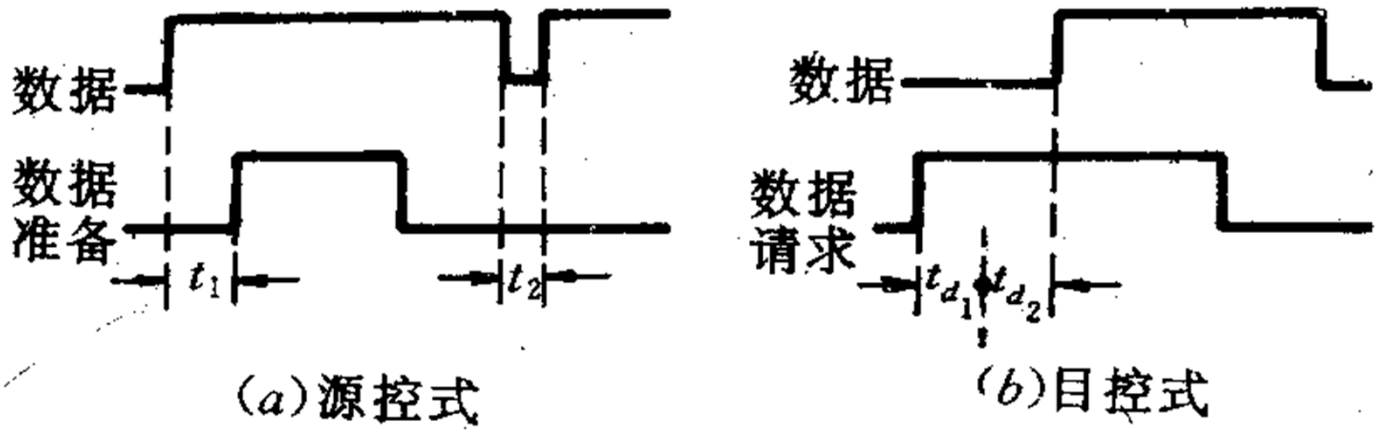


图 异步单向控制通信

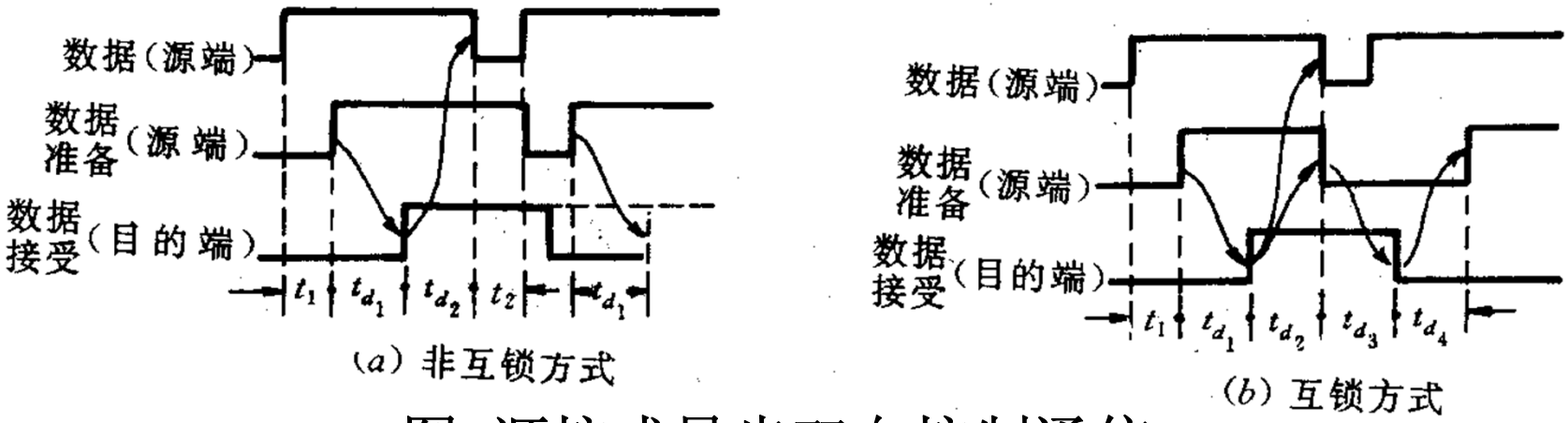


图 源控式异步双向控制通信

总线仲裁

- 总线判优。
- 当采用非专用总线时，由于可能发生多个设备或部件同时申请使用总线，就得有总线控制机构来按照某种优先次序裁决，保证在同一时间内只能有一个高优先级的申请者取得对总线的使用权。
- 根据总线控制部件的位置，控制方式可以分成集中式与分布式两类。

1. 集中式仲裁

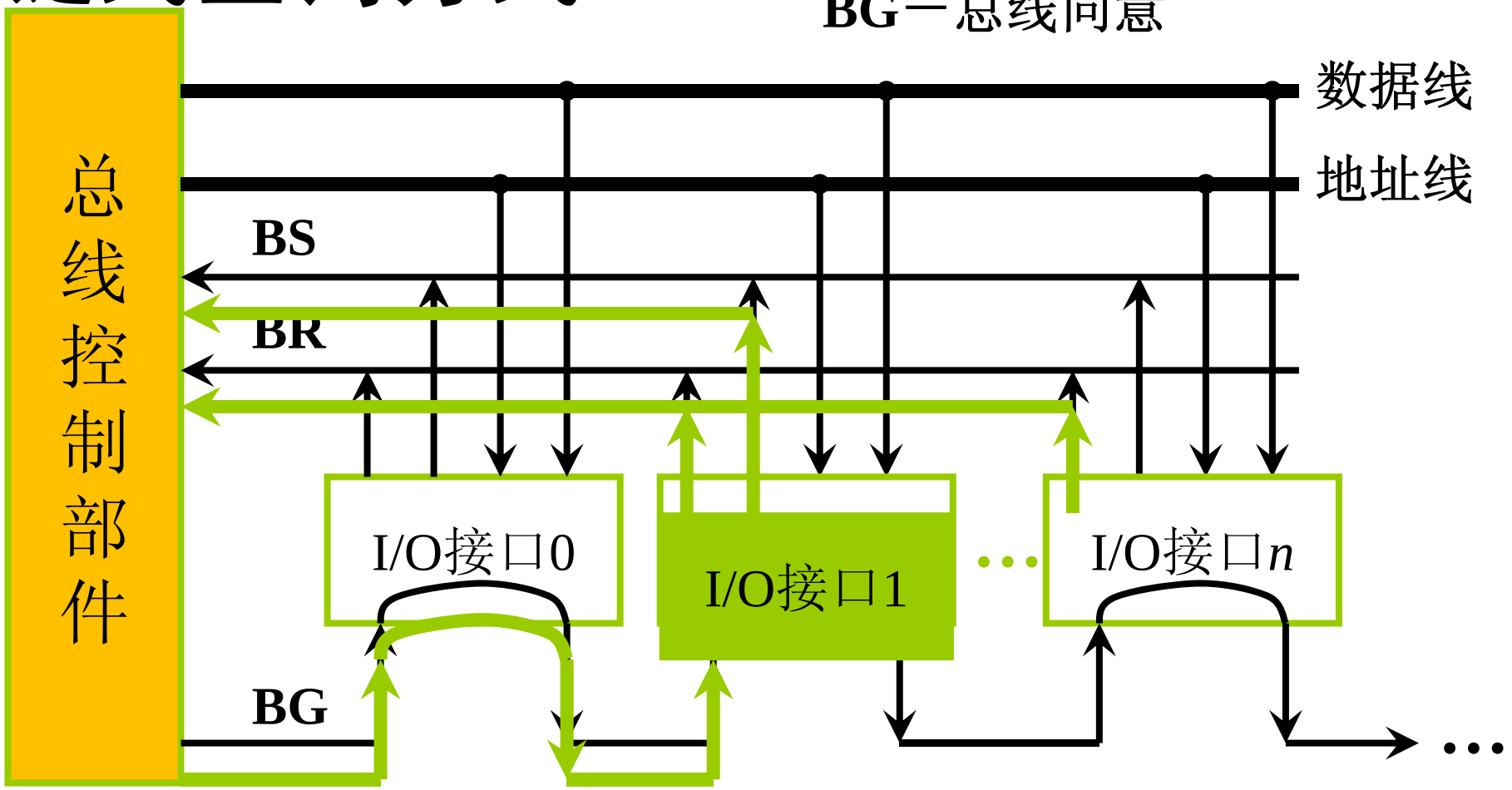
- 如果总线控制逻辑基本上集中放在一起的，不论是放在连接到总线的一个部件中，还是放在单独的硬件中，都称为集中式控制。
- 三种常见优先权仲裁方式：
  - 链式查询方式
  - 计数器定时查询方式
  - 独立请求方式

链式查询方式

- 链式查询方式需要有三根控制线：
  - **总线忙BS：** 该信号有效时，表示总线正被某外设使用。
  - **总线请求BR：** 该信号有效时，表示至少有一个外设请求使用总线。
  - **总线批准BG：** 该信号有效时，表示总线控制部件响应了外设的总线请求。

链式查询方式

BS－总线忙
BR－总线请求
BG－总线同意



- 控制线3根： 总线状态BS，总线请求BR，总线授权BG
- 仲裁过程：监控总线状态，发总线请求(BR)，等待总线授权；截获授权(BG)，置总线状态(BS)，使用总线， 释放总线。
- 响应慢；优先级固定；单点故障敏感； 饥饿现象；

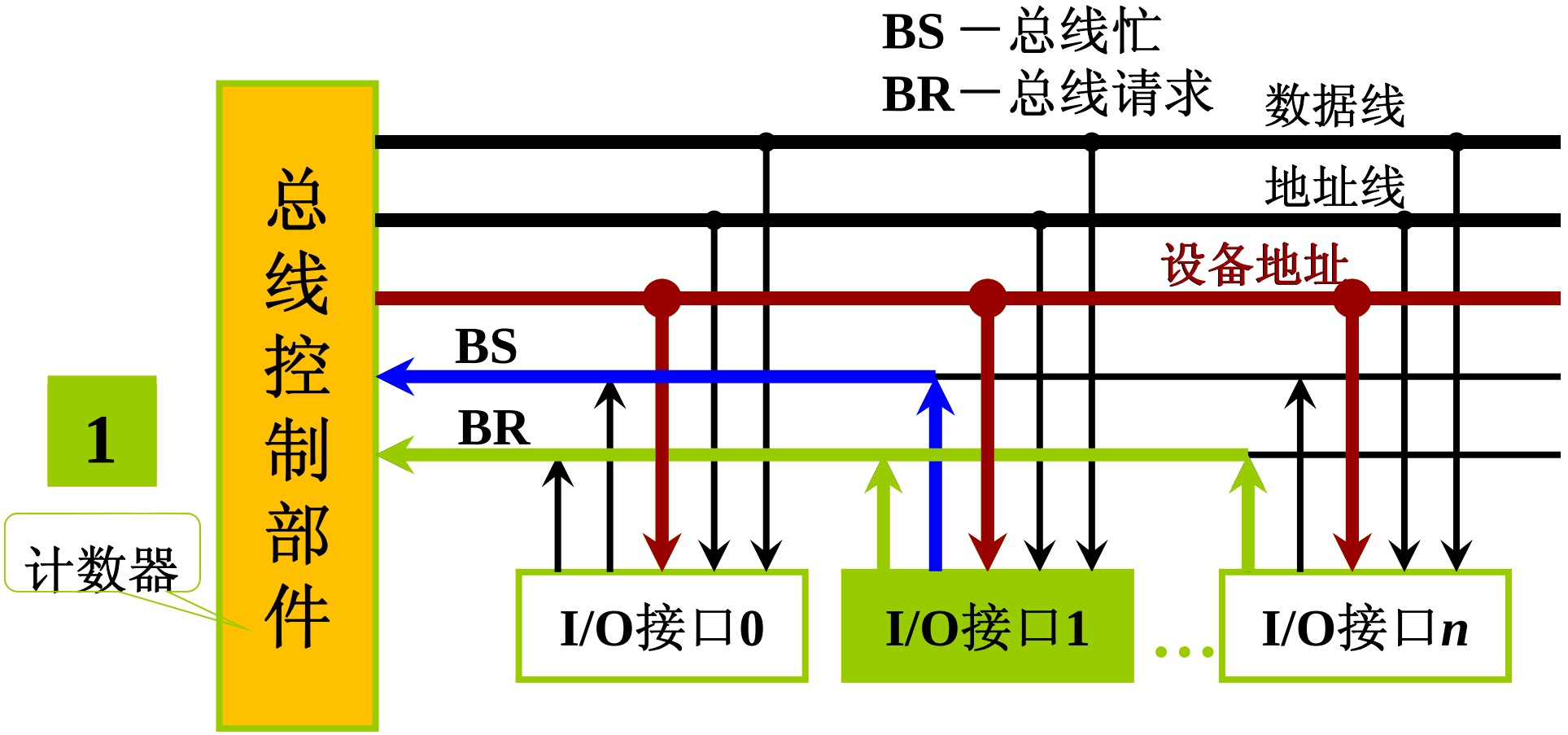
计数器定时查询方式

- 总线上任何设备要求使用总线时，都通过BR线发出总线请求。
- 总线控制器接到总线请求信号后，在BS线为0的情况下其设备地址计数器开始计数，计数值通过一组设备地址线发向各设备。
- 每个外设接口都有一个设备地址判别电路，当设备地址线上的计数值与请求使用总线的设备地址一致时，该设备就获得了总线使用，并置BS线为1。

计数器定时查询方式

- 总线控制器根据检测到BS信号时的设备地址就知道当前哪个设备使用了总线。
- 每次设备地址计数可以从0开始，也可从上次计数的中止点开始。
  - 如果从0开始，各设备的优先次序与链式查询相同，优先级的顺序是固定的。
  - 如果从中止点开始，则每个设备使用总线的优先级别是相等的。计数器的初值也可以用程序来设置，这就可以方便地改变优先次序。

计数器定时查询方式

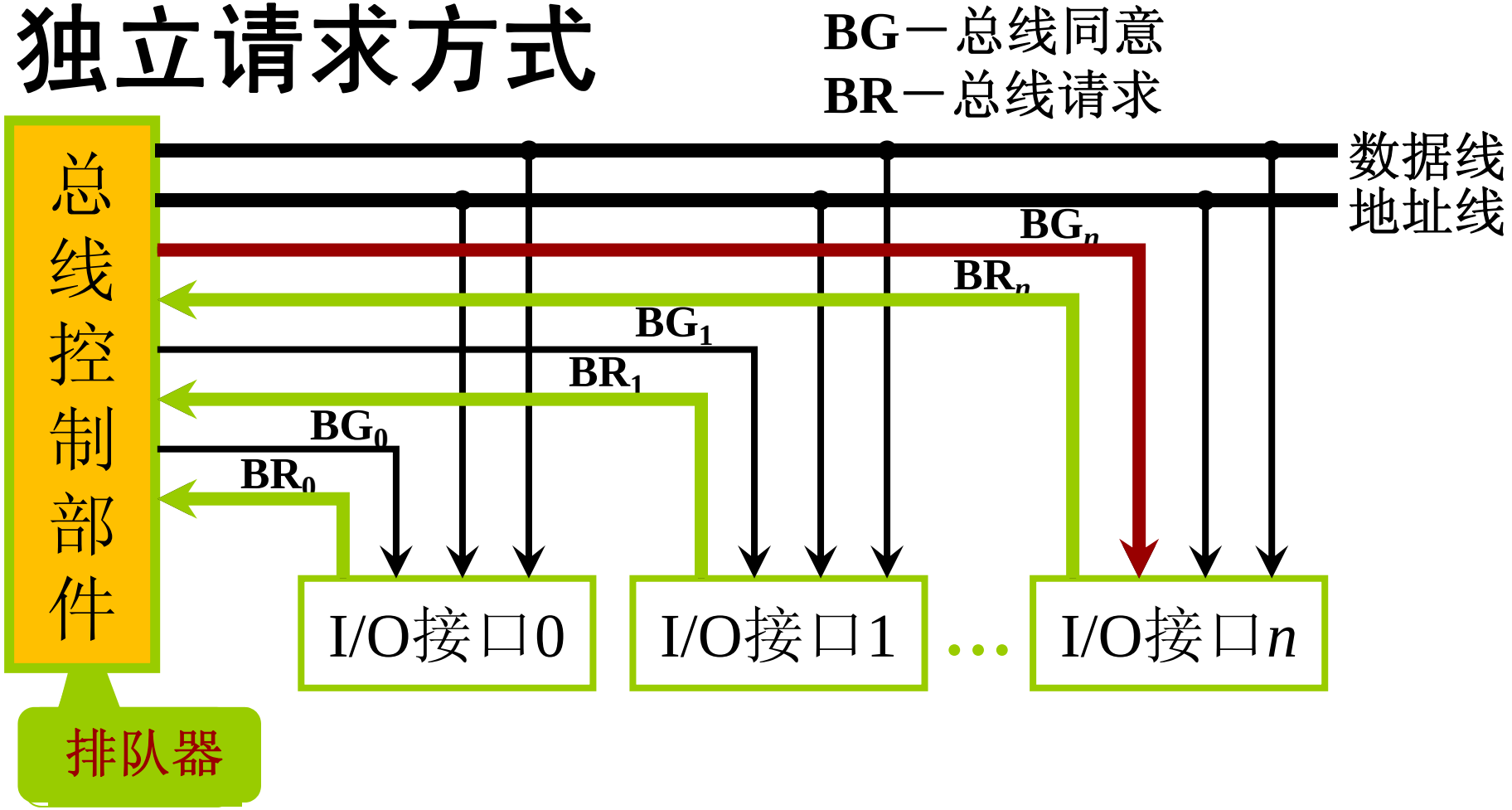


- 控制线＝2＋[$\log_2 n$]根，总线状态BS、总线请求BR、地址计数。
- 仲裁过程：总线授权通过设备地址计数来判别。
- 响应慢；优先级可变化；故障不敏感；扩展困难，线数量多。

独立请求方式

- 每个设备均有独立的总线请求线BR_i和总线回答线BG_i。当设备要求使用总线时，便发出总线请求信号BR_i。
- 总线控制器对所有的总线请求进行优先级排队，响应级别最高的请求，并向该设备发出总线回答信号BG_i。
- 得到响应的设备将占用总线进行传输。

独立请求方式



- 控制线2n根，总线请求BR、总线授权BG，无总线状态信号。
- 仲裁过程：总线请求，等待总线授权。
- 响应快；优先级可灵活变化； 故障不敏感；扩展容易。

独立请求方式

- 响应时间快，但控制线数量多。
  - 在链式查询中仅用两根线确定总线使用权属于那个设备；在计数查询中大致用$\log_2 n$根线，其中n是允许接入的最大设备数。而独立请求方式需采用2n根线。
- 对优先次序的控制也是相当灵活的。
  - 它可以预先固定，如让BR0优先级最高、BR1次之、...、BRn最低；或者通过程序来改变优先次序；或者通过屏蔽不响应来自与当前处理无关的设备的请求。
- 控制器复杂。

三种方式比较

	链式查询方式	计数器定时查询	独立请求方式
控制线	BS、BR、BG 共3根	BS、BR、log2n 共 2+⌈log2n⌉	n组（BR、BG） 共2n根
响应速度	慢	慢	快
优先级	优先级固定	可作适当变化	可作灵活的变化
故障敏感度	非常敏感	不敏感	不敏感
扩展方式	容易	难	容易

2. 分布式仲裁

- 分布仲裁方式不需要中央仲裁器，即总线控制逻辑分散在连接于总线上的各个部件或设备中
- 连接到总线上的主方可以启动一个总线周期，而从方只能响应主方的请求。
- 每次总线操作，只能有一个主方占用总线控制权，但同一时间里可以有一个或多个从方。对多个主设备提出的占用总线请求，一般采用优先级或公平策略进行仲裁。

2. 分布式仲裁

- 每个潜在的主方功能模块都有自己的仲裁号和仲裁器。当它们有总线请求时，把它们惟一的仲裁号发送到共享的仲裁总线上；
- 每个仲裁器将仲裁总线上得到的号与自己的号进行比较：
  - 如果仲裁总线上的号大，则它的总线请求不予响应，并撤消它的仲裁号；
  - 最后，获胜者的仲裁号保留在仲裁总线上；
- 显然，分布式仲裁是以优先级仲裁策略（或公平策略）为基础的。

2. 分布式仲裁

- 三种常见仲裁方式：
  - 自举分布式
  - 冲突检测分布式
  - 并行竞争分布式

自举分布式

- 每个设备的优先级固定，需要请求总线控制权的设备在各自对应的总线请求线上送出请求信号。

- 在总线仲裁期间，每个设备通过取回的信息能够检测出其他比自己优先级高的设备是否发出了总线请求。
- 如果没有，则立即使用总线，并通过总线忙信号阻止其他设备使用总线；
- 如果一个设备在发出总线请求的同时，检测到其他优先级更高的设备也请求使用总线，则本设备不能马上使用总线。

冲突检测分布式

- 当某个设备要使用总线时，首先检查是否有其他设备正在使用总线，如果没有，则它就置总线忙，并直接使用总线。

- 若两个设备同时检测到总线空闲，那它们可能都会立即使用总线，从而发生冲突。
- 因此，每个设备在使用过程中，会侦听总线以检测是否发生冲突，当发生冲突，两个设备都会停止传输，延迟一个随机时间之后再重新使用总线，以避免冲突
- 一般用在网络通信总线上，Ethernet CSMA/CD 就是使用该方案

并行竞争分布式

- 这是一种较复杂、但有效的裁决方案。
- 基本思想：
  - 总线上的每个设备都有一个唯一的仲裁号，需要使用总线的主控设备把自己的仲裁号发送到仲裁线上，这个仲裁号将用在并行竞争算法中。
  - 每个设备根据仲裁算法决定在一定时间段后占用总线还是撤销仲裁号。

3.4 通道处理机

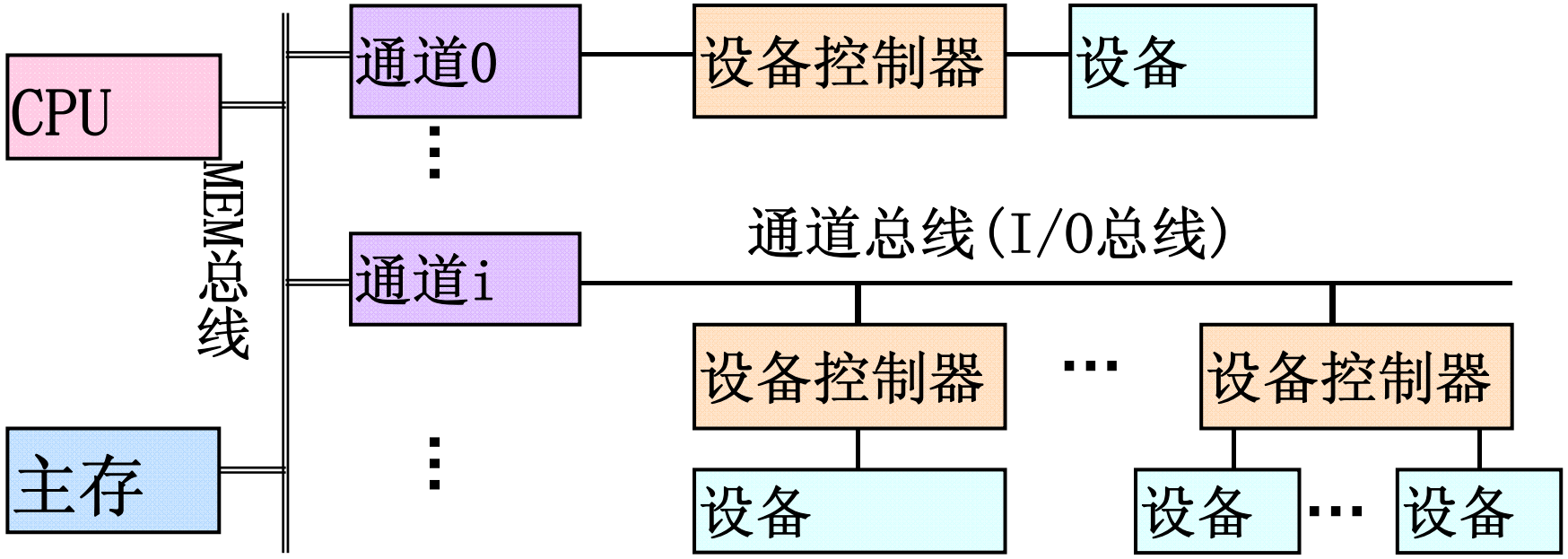
- 由IBM公司首先提出的一种I/O处理方式。后被广泛使用
  - 例如IBM 360和370系列机

目标：进一步减少CPU的传送控制工作负担。

通道：具有特殊功能的处理器，执行程序实现I/O传送控制。

3.4 通道处理机

- **通道方式：** 通道控制的主存-外设间直接数据传送方式。



3.4.1 通道的作用和功能

- 作用
  - 减轻CPU处理输入输出的负担；
  - 高效管理设备，使设备并行工作，提高利用率；

3.4.1 通道的作用和功能

- 特点：
  - 它拥有通道指令和通道程序
    - 通道指令为I/O设备规定一定的动作。通道通过执行通道指令对外设进行控制，如发出读、写命令等；
    - 通道程序由多条通道指令链接而成，存放在对应该通道的主存缓冲区中。通道通过执行通道程序来控制输入输出。通道程序可以和CPU程序并行工作。

3.4.1 通道的作用和功能

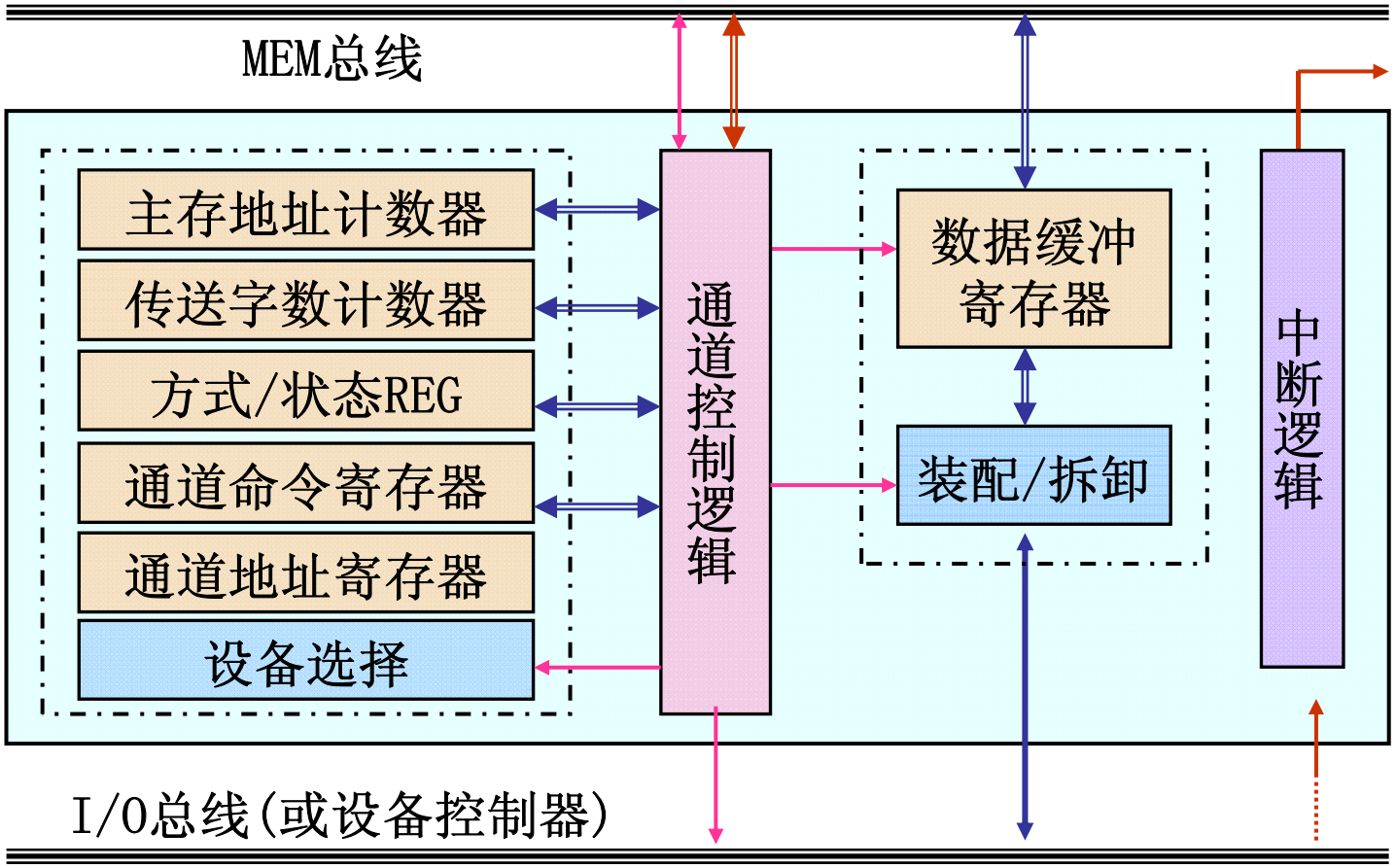
- 特点：
  - 可与CPU并行工作
  - 由通道统一管理外设
    - 替代CPU对多个设备的信息传输进行分时管理；
    - 实现字与字节的装配和拆卸；
    - 向CPU报告设备/控制器的状态及状态分析结果；
    - 对输入输出系统出现的各种情况进行处理等。

3.4.1 通道的作用和功能

- 主要功能：
  - 接收CPU的I/O指令，与指定设备联系；
  - 执行通道程序（指令），控制I/O设备及传送；
  - 组织与控制I/O传送，数据处理
    - 给出交换信息主存地址和长度；
    - 给出外设有关地址；
    - 装配和拆卸信息，格式变换；
  - 向CPU报告状态及状态分析结果；
  - 向CPU发出中断请求（外设、通道本身）；

3.4.1 通道的作用和功能

- 硬件组成：寄存器和控制逻辑



3.4.2 通道工作过程

- 大多数计算机系统是由操作系统分配、调度I/O设备，并进行具体的I/O处理，而不是由用户直接安排输入输出。
- CPU控制外设操作的I/O指令被定义为管态指令。
- 用户在目态程序使用广义指令请求I/O服务。
- 广义指令由访管指令和若干参数组成。
- 访管指令为目态指令。

3.4.2 通道工作过程

- 当目态程序执行到要求输入输出的访管指令后，产生访管中断，CPU相应中断，转向管理程序入口，进入管态(目态➔管态)；
- 管理程序根据广义指令所提供的参数，如设备号、主存起始地址、信息长度等编制通道程序，并将其存放在某通道对应的通道缓冲区中，将通道程序入口地址置入通道地址单元；
- 之后，管理程序启动“启动I/O指令”，开始通道的开始选择设备期。

3.4.2 通道工作过程

- “启动I/O指令”是主要的I/O指令，属于管态指令。
- 通道启动后，按通道程序组织I/O操作，进入通道的数据传送期，开始通道与设备之间的数据传送；
- 与此同时，CPU退出管态，返回到目态程序继续执行(管态➔目态)。

3.4.2 通道工作过程

- 通道程序执行完无链通道指令后，传送完成，转入通道的数据传送结束期，向CPU发出I/O中断请求；
  - 当然，若出现错误、故障等异常，它也向CPU发出I/O中断请求。
- CPU相应中断请求，进入管态，调用管理程序对I/O中断进行处理(目态➔管态)；
- 若属于正常结束，就进行必要的登记计费；若属于故障、错误，则进行相应的处理。然后再转回目态，继续目态程序的执行(管态➔目态)。

3.4.2 通道工作过程

- 通道完成一次数据输入输出的过程需三步：
  - (1) 传送准备。在用户程序中使用访管指令进入管理程序，由CPU通过管理程序组织一个通道程序，并启动通道。
  - (2) 数据传送。通道处理机执行通道程序，完成指定的数据输入输出工作。
  - (3) 传送结束。通道程序结束后第二次调用管理程序对输入输出请求进行处理。

每完成一次输入输出工作，CPU只需要两次调用管理程序，大大减少了对用户程序的打扰。

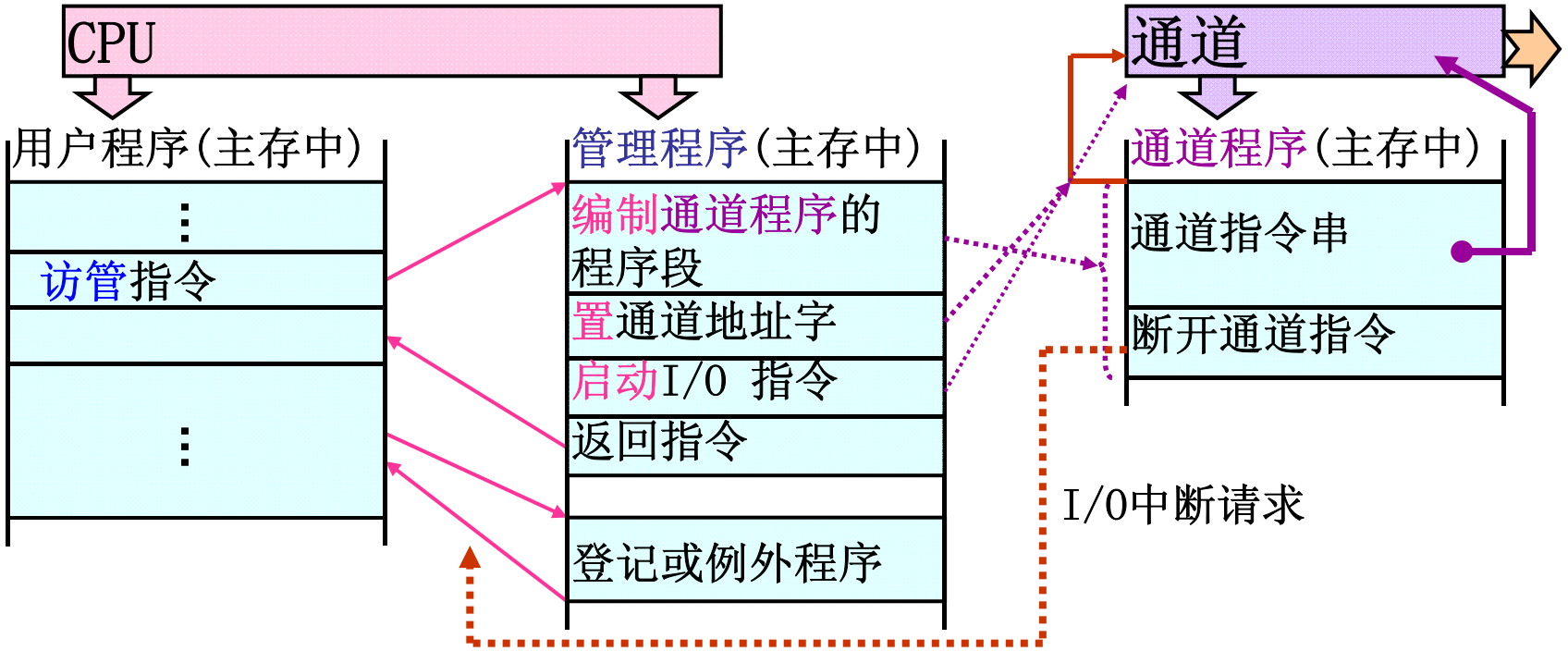
3.4.2 通道工作过程

*用户程序的访管指令，调用I/O管理程序

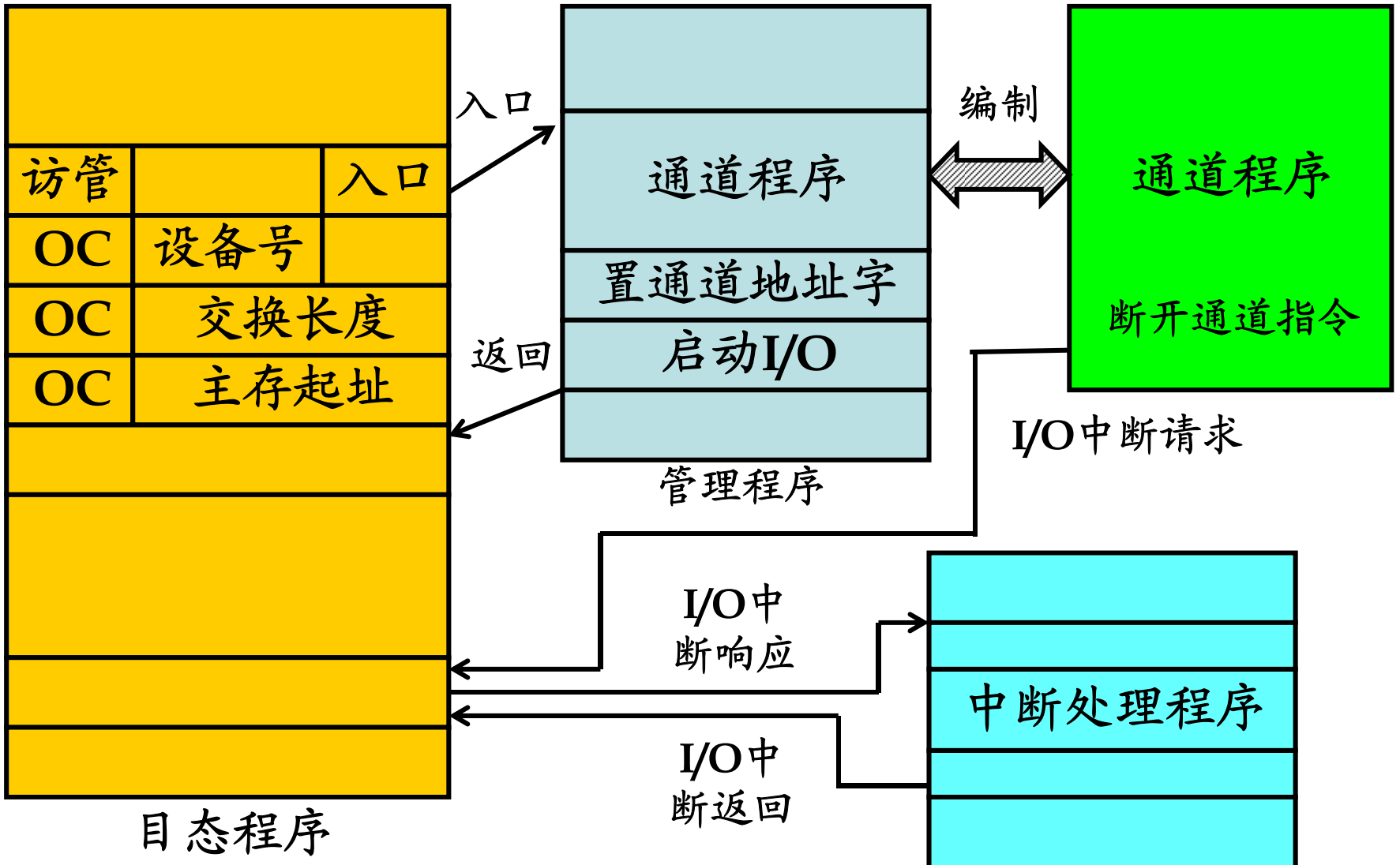
访管指令格式

操作码	通道号	设备号	操作	交换长度	主存首址

*I/O管理程序实现I/O传送需求表示、启动通道功能

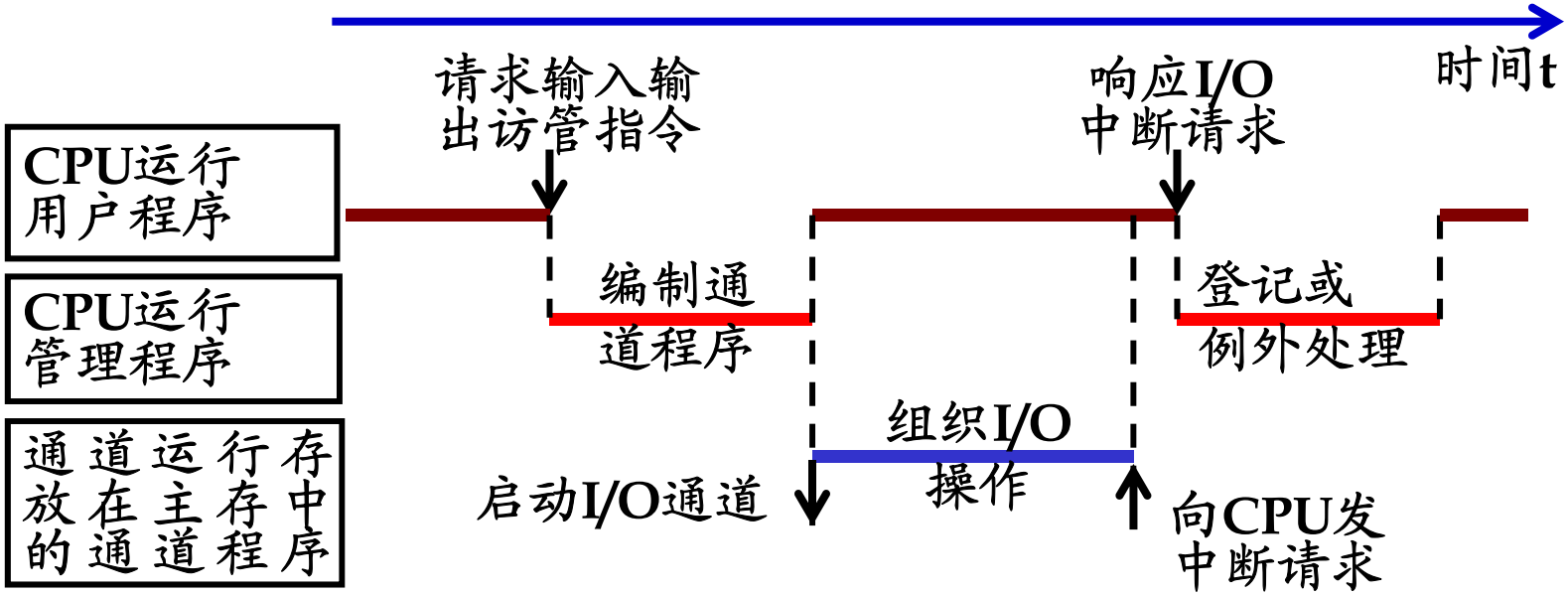


3.4.2 通道工作过程


访管
入口
OC
设备号
OC
交换长度
OC
主存起址
目态程序
入口
返回
通道程序
置通道地址字
启动I/O
管理程序
编制
通道程序
断开通道指令
I/O中断请求
I/O中
断响应
I/O中
断返回
中断处理程序

3.4.2 通道工作过程

- 可以看出：完成一次I/O操作只需要两次进管：
  - 一次是处理要求输入输出的访管中断的时候；
  - 一次是处理输入输出结束的I/O中断的时候。



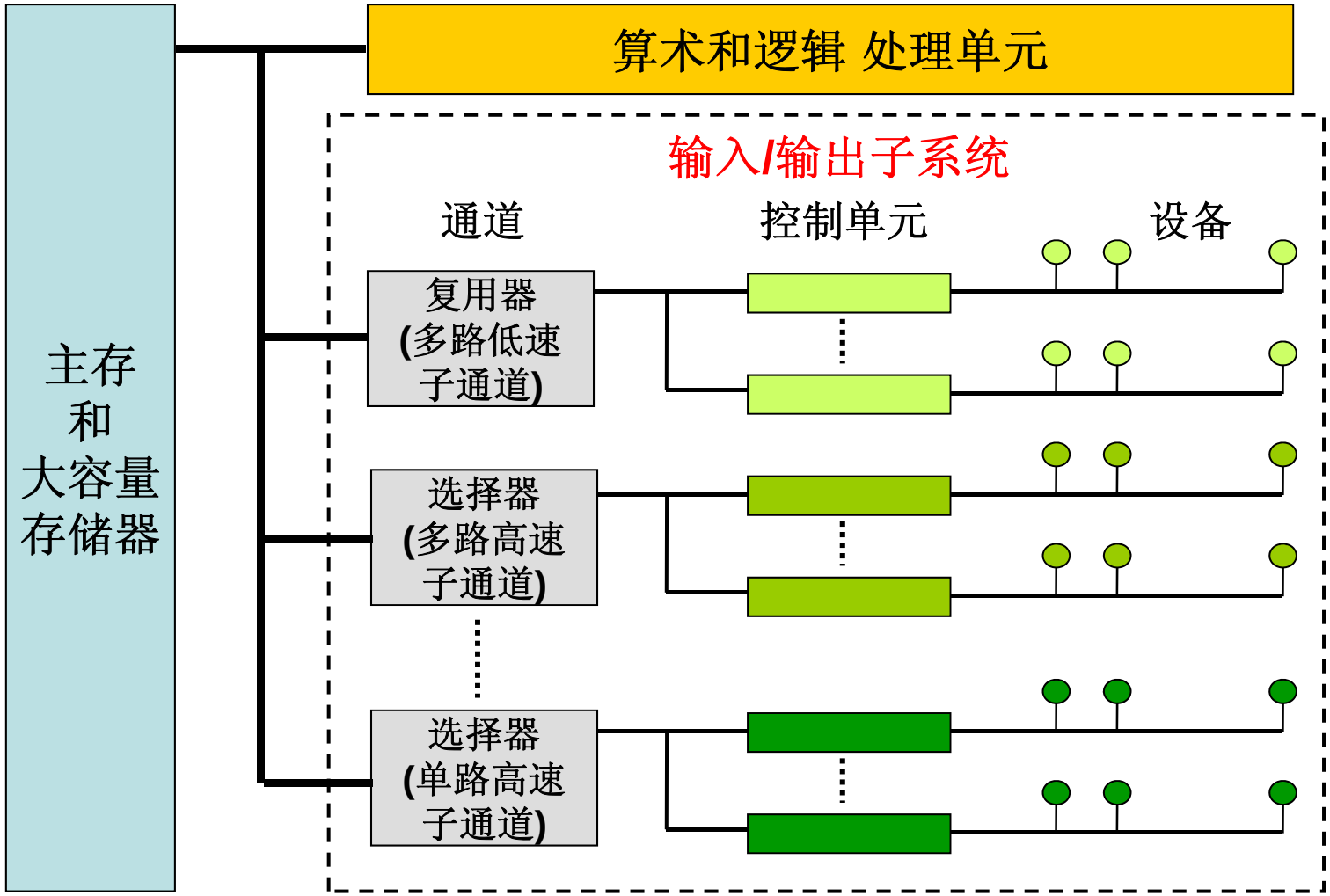
3.4.2 通道工作过程

- 通道的工作分为三个阶段：
  - 传送准备期 (CPU进入管态)
    - 选择通道、子通道和通道指令
    - 选择控制器和设备
    - 启动I/O
  - 数据传送期 (CPU进入目态)
  - 传送结束期 (CPU进入管态)

3.4.3 通道的类型

- 按信息传送的方式，分为三种类型：
  - 字节多路通道
  - 数组多路通道
  - 选择多路通道
- 它们的速度和容量是不同的。

3.4.3 通道的类型



IBM 360/370系列机的功能结构

1. 字节多路通道

- 一种简单的共享通道。
- 适用于连接大量低速外设。
  - 低速设备传送一个字节的时间很短，但字节之间的间隔时间很长。
- 每次传送一个字节，采用字节交叉方式轮流为每个低速设备服务。
- 可以有多个子通道，各子通道能独立执行通道指令，各子通道能“并行”操作。

1. 字节多路通道

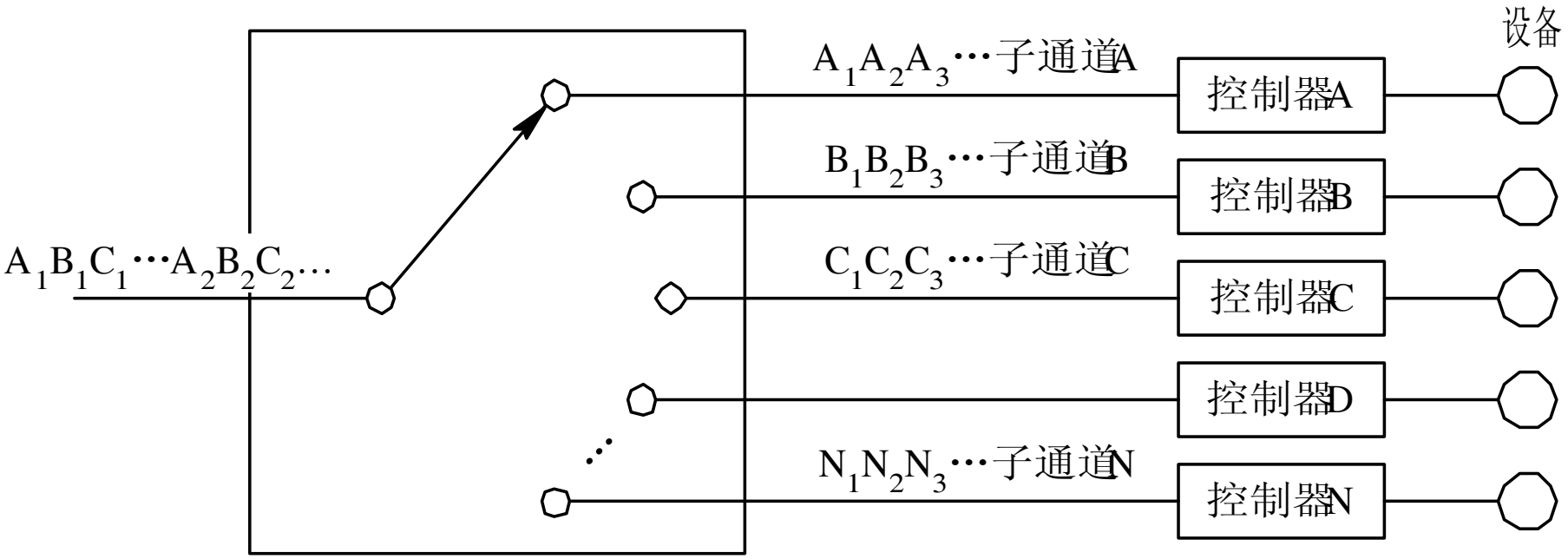
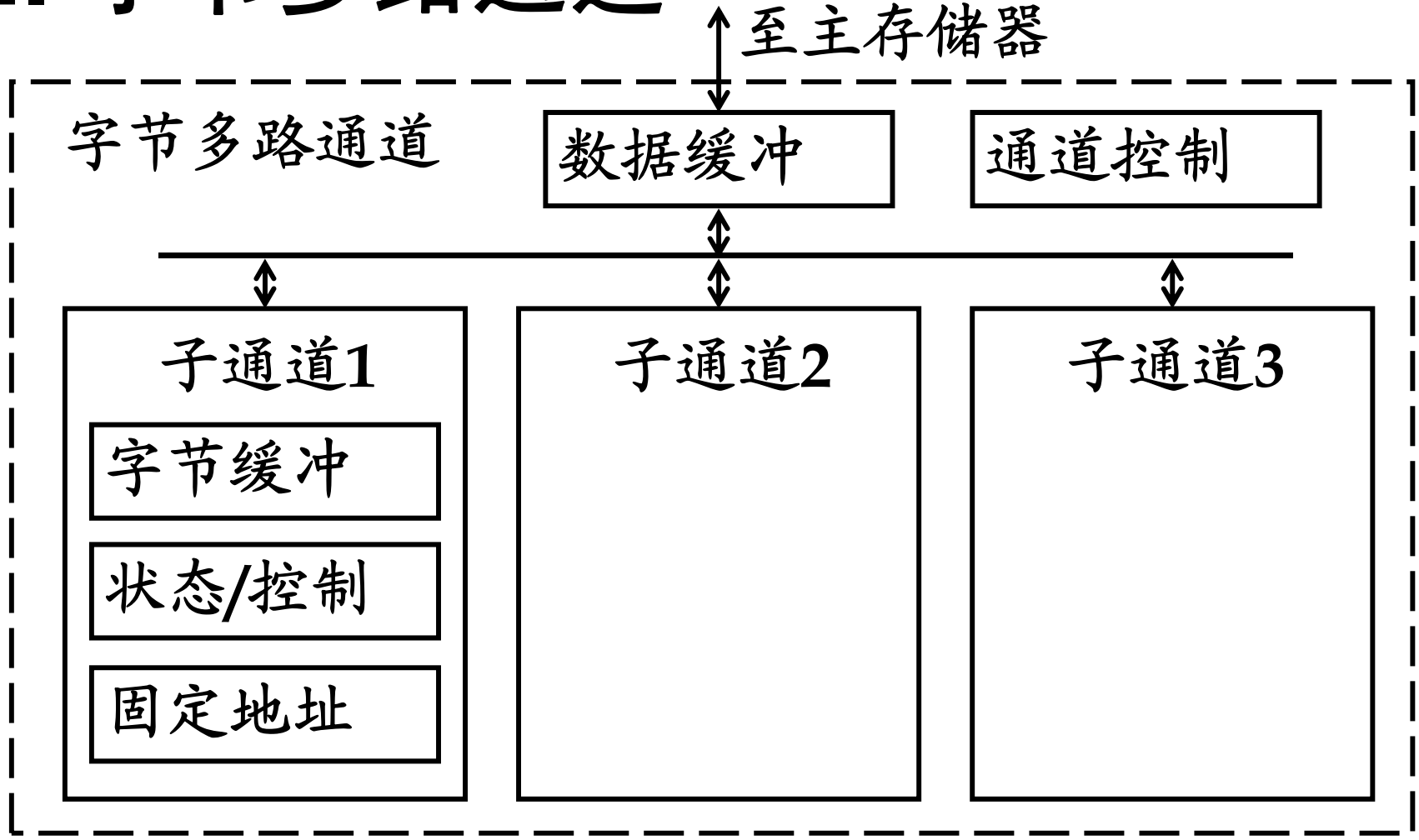


图 字节多路通道的工作原理

1. 字节多路通道



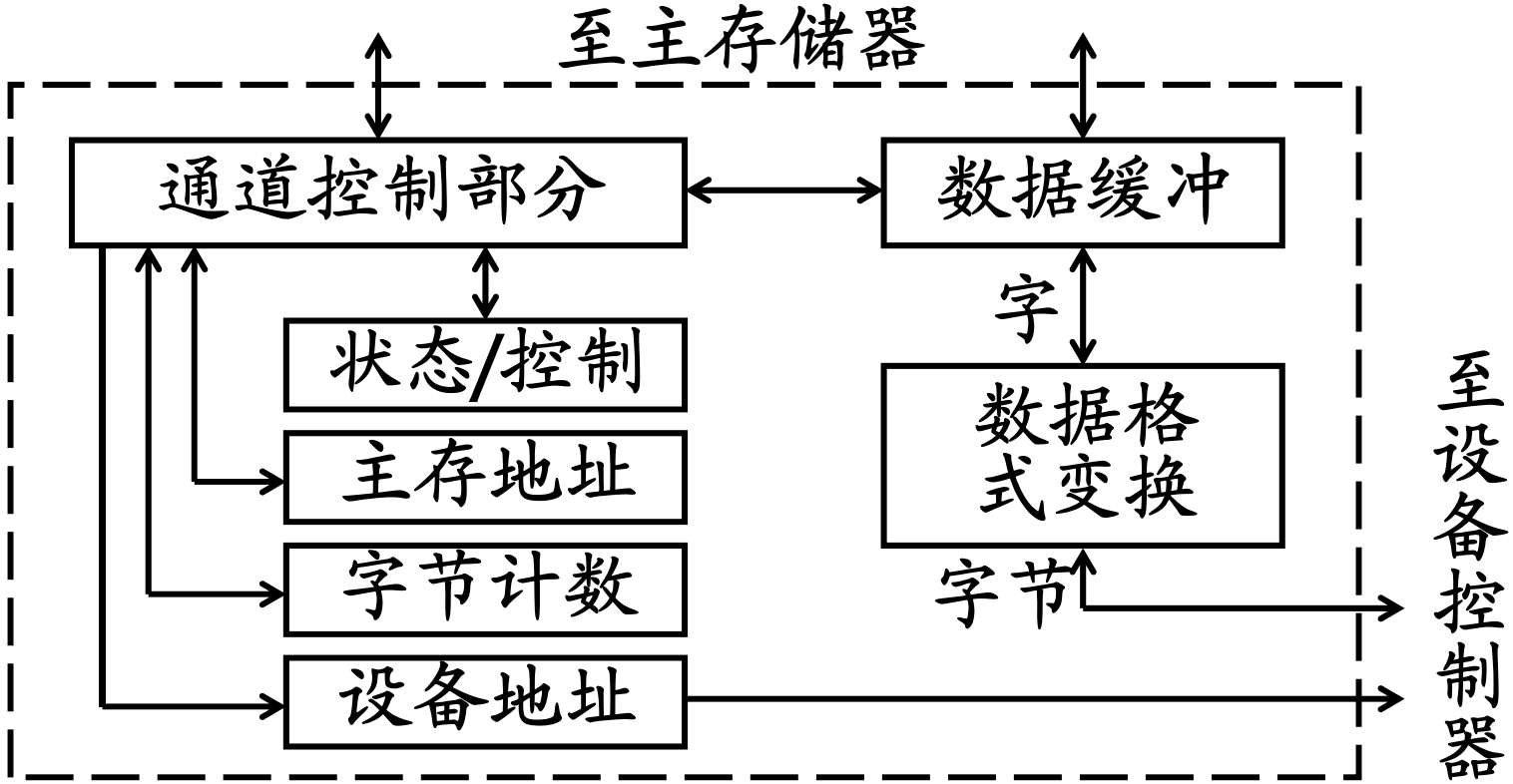
字节多路通道的结构

2. 选择多路通道

- 又称为高速通道。
- 适合于连接优先级较高的磁盘一类的高速外设。
- 在物理上它也可以连接多个设备，但这些设备不能同时工作。在一段时间内通道只能选择一台设备进行数据传送，此时该设备能独占整个通道。
- 每次传送一个变长数据块。

2. 选择多路通道

每个选择通道只有一个以成组方式工作的子通道，逐个为多台高速外围设备服务。



选择通道的结构

3. 数组多路通道

- 适用于为类似磁盘等高速外设服务。
  - 这类设备传送速率很高，但传送开始前的寻址等辅助操作时间长。
- 当某设备进行数据传送时，通道只为该设备服务。当设备在执行辅助操作（如磁头移动等）时，通道暂时断开与这个设备的连接，挂起该设备的通道程序，去为其他设备服务。
- 每次传送一个定长数据块，采用成组交叉方式工作。

3. 数组多路通道

- 数组多路通道之所以能够并行地为多个高速外围设备服务，是因为这些高速外围设备并不能在整个数据输入输出时间内单独利用通道的全部传输能力。
- 可以有多个子通道，所有子通道能分时共享输入输出通路，具有多路并行操作能力和很高的数据传送速率。

3.4.4 通道流量分析

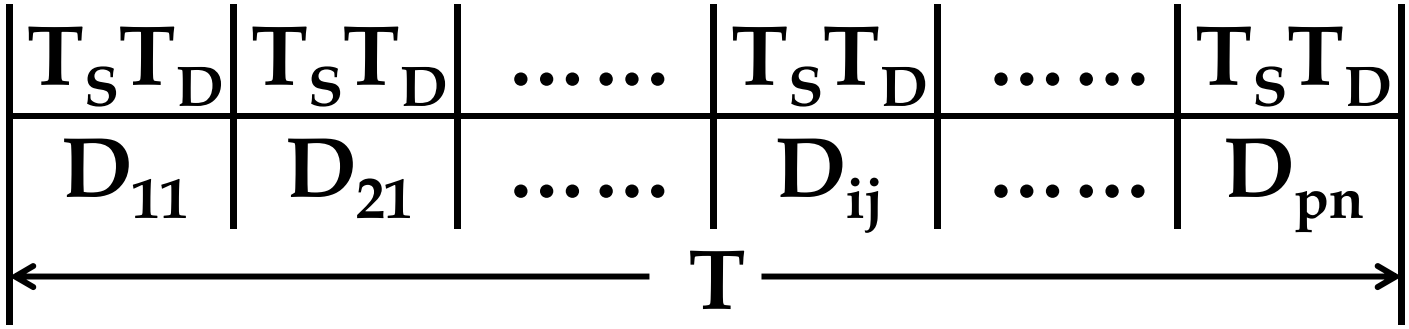
- 通道流量
  - 在数据传送期间，单位时间内传送的字节数（BPS）。
  - 又称为通道吞吐率、通道数据传输率等。
- 常用两种：
  - 通道极限流量
  - 实际流量

3.4.4 通道流量分析

- **通道极限流量**
  - 通道所能达到的最大通道流量
  - 它与通道的工作方式、数据传送期内的选择一次设备的时间T_S以及传送一个字节的时间T_D有关
- **实际流量**
  - 通道上挂接设备后所有设备要求的通道流量。

字节多路通道流量分析

- 字节多路通道的数据传送过程
  - 分时为多台低速和中速外设服务的，每选择一台设备只传送一个字节。在有P台设备同时连接到一个字节多路通道上时，它们的数据传送过程如图所示。



T_S=设备选择时间，T_D=传送一个字节的时间，P=通道连接的设备数，
n=每个设备传送的字节数，D_{ij}=第 i 台设备传送的第 j 个数据，
T=完成数据传送所需要的时间

字节多路通道流量分析

- P 台设备每台传送 n 个数据总共所需的时间为：

$$T_{BYTE} = (T_S + T_D) \times P \times n$$

- 极限流量：

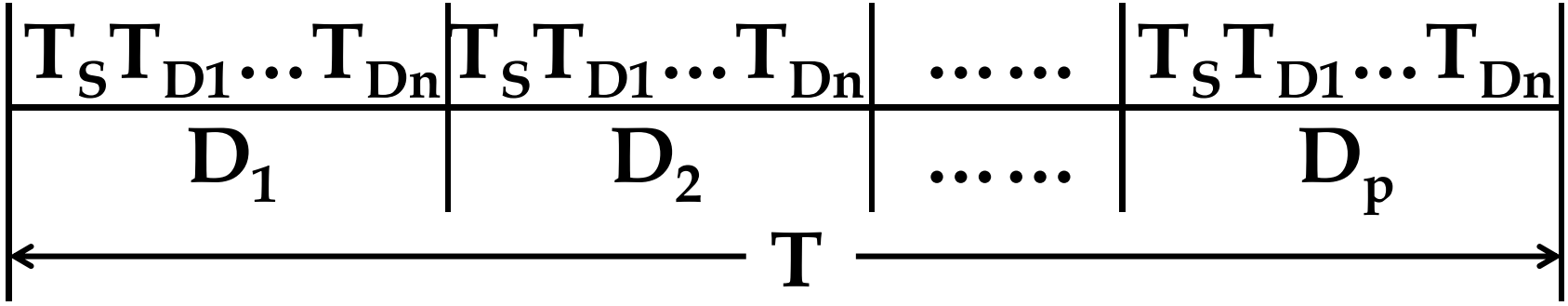
$$f_{max \bullet byte} = \frac{P \times n}{(T_S + T_D) \times P \times n} = \frac{1}{T_S + T_D}$$

- 实际流量：P 台设备传输率之和。

$$f_{byte} = \sum_{i=1}^{P} f_i$$

选择通道流量分析

- 选择通道的数据传送过程
  - 在一段时间内只能单独为一台高速外设服务，当这台设备的数据传送工作**全部**完成后，通道才能为另一台设备服务（**每选择一台设备就把n个字节全部传送完**）。



$$T_{D1} = T_{D2} = \cdots = T_{Dn} = T_D$$

选择通道流量分析

- P 台设备每台传送 n 个数据总共所需的时间为：

$$T_{SELECT} = (\frac{T_S}{n} + T_D) \times P \times n$$

- 极限流量：

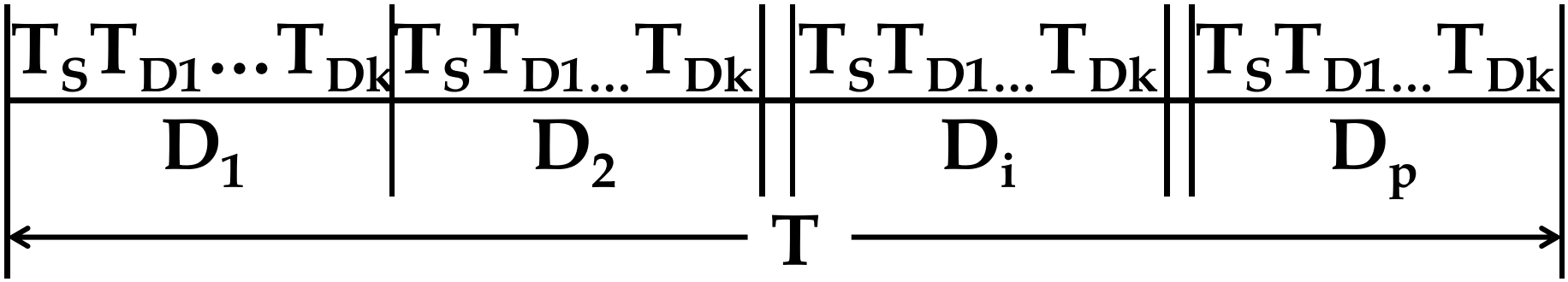
$$f_{max \bullet select} = \frac{P \times n}{(\frac{T_S}{n} + T_D) \times P \times n} = \frac{n}{T_S + nT_D}$$

- 实际流量：P 台设备中传输率最高者。

$$f_{select} = \max_{i=1}^{P} f_i$$

数组多路通道流量分析

- 数组多路通道的数据传送过程
  - 每连接一台高速设备，就传送一个定长数据块。传送完成后，又与另一台高速设备连接，再传送一个数据块（每选择一台设备只传送定长 k 个字节）。



数组多路通道流量分析

- P 台设备每台传送 n 个数据总共所需的时间为：

$$T_{BLOCK} = (\frac{T_S}{k} + T_D) \times P \times n$$

- 极限流量：

$$f_{max \bullet block} = \frac{P \times n}{(\frac{T_S}{k} + T_D) \times P \times n} = \frac{k}{T_S + kT_D}$$

- 实际流量：P 台设备中传输率最高者。

$$f_{block} = \max_{i=1}^{P} f_i$$

3.4.4 通道流量分析

- 第 j 个字节多路通道的实际最大流量（求和）

$$f_{byte \bullet j} = \sum_{i=1}^{p_j} f_{i \bullet j}$$

- 第 j 个选择通道或数组多路通道的实际最大流量（求和）

$$f_{select \bullet j} = \max_{i=1}^{p_j} f_{i \bullet j}$$

$$f_{block \bullet j} = \max_{i=1}^{p_j} f_{i \bullet j}$$

$f_{i,j}$ — 第 j 号通道上第 i 个设备的数据传输率

P_j — 第 j 号通道上的设备数

3.4.4 通道流量分析

- 流量设计的基本原则
  - 如果I/O系统有m个通道，其中$1\sim m_1$为字节多路通道，$m_1+1\sim m_2$为数组多路通道，$m_2+1\sim m$为选择多路通道，则

该I/O系统的通道极限流量为：

$$f_{\max}=\sum_{j=1}^{m_1} f_{\max\bullet byte\bullet j}+\sum_{j=m_1+1}^{m_2} f_{\max\bullet block\bullet j}+\sum_{j=m_2+1}^{m} f_{\max\bullet select\bullet j}$$

该I/O系统的实际最大流量为：

$$f=\sum_{j=1}^{m_1}\sum_{i=1}^{pj} f_{i\bullet j}+\sum_{j=m_1+1}^{m_2}\max_{i=1}^{pj} f_{i\bullet j}+\sum_{j=m_2+1}^{m}\max_{i=1}^{pj} f_{i\bullet j}$$

3.4.4 通道流量分析

- 通道设计的基本原则
  - 为保证第 j 号通道上的设备在满负荷的最坏情况下都不丢失数据，必须满足以下**最基本原则**：

原则1：通道的实际流量不超过通道的极限流量。

$$f_{byte \bullet j} < f_{\max \bullet byte \bullet j}$$

$$f_{block \bullet j} < f_{\max \bullet block \bullet j}$$

$$f_{select \bullet j} < f_{\max \bullet select \bullet j}$$

必须满足：

$$f_{\max} \geq f$$

3.4.4 通道流量分析

- 可以用左右两边的差值来衡量I/O系统流量的利用率。
  - 差值越小，利用率越高
  - 当两边相等时，通道处于满负荷工作状态
- f_{max}也是I/O系统对主存频宽B_m的要求

原则2：一般安排传送速率高的设备请求具有较高的响应优先级。

3.4.4 通道流量分析

- 例1：

	子通道	挂接设备	请求间隔
字节多路通道	子通道1	0号高速打印机	25μs
	子通道2	1号高速打印机	25μs
	子通道3	0号打印机 1号打印机 0号光电输入机	150μs 150μs 800μs

3.4.4 通道流量分析

- 例1：设计该字节多路通道的极限流量。画出该通道响应和处理设备请求的时间关系图（假设首次请求时，0号高速打印机比其他设备晚 5μs 之后提出请求）。

3.4.4 通道流量分析

- 例1：解
  - 通道的实际流量为：

$$f_{byte \bullet j} = \sum_{i=1}^{5} f_{i \bullet j} = \frac{1}{25} + \frac{1}{25} + \left(\frac{1}{150} + \frac{1}{150} + \frac{1}{800} \right) \approx 0.095\text{MB/s}$$

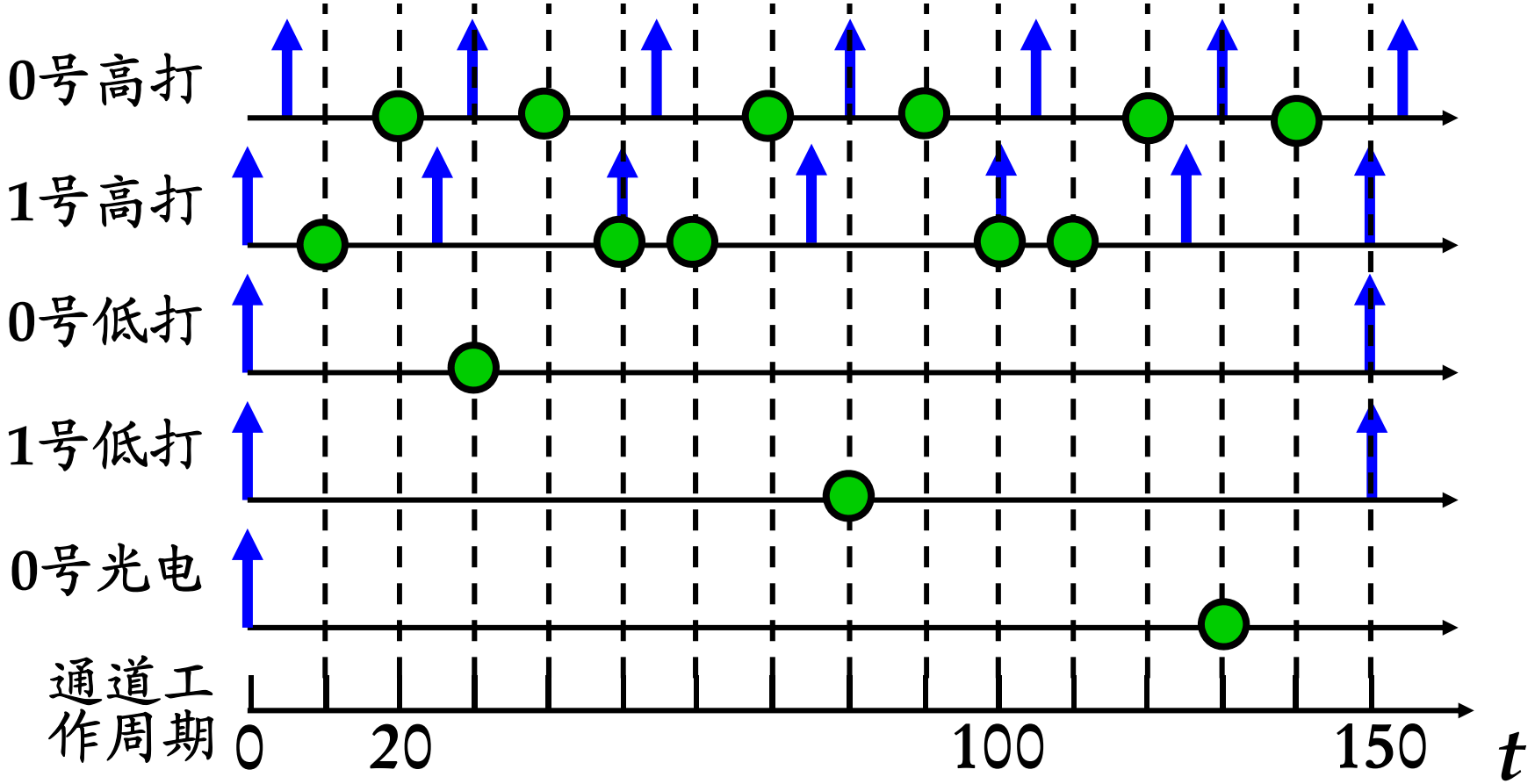
  - 通道的极限流量可以设计为：

$$f_{\max \bullet byte \bullet j} = 0.1\,\text{MB/s} > f_{byte \bullet j}$$

  - 通道的工作周期为：$t = (T_S + T_D) = 10\mu s$

3.4.4 通道流量分析

- 例1 解：时间示意图如下



3.4.4 通道流量分析

- 例2：一个字节多路通道连接D1、D2、D3、D4、D5共5台设备，这些设备分别每10μs、30μs、30μs、50μs和75μs向通道发出数据传送的服务请求，请回答下列问题：

3.4.4 通道流量分析

- 例2（续）：
  - (1) 计算这个字节多路通道的实际流量。
  - (2) 如果该字节多路通道的流量正好等于实际流量，画出5台设备同时提出服务请求时该通道的时间关系图，并计算处理完各台设备第一次数据传送请求的时刻。
  - (3) 从时间关系图上发现什么问题？如何解决这个问题？

3.4.4 通道流量的分析

- 例2 解：
  - (1)通道的实际最大流量为：

$$f_{BYTE}=\left(\frac{1}{10}+\frac{1}{30}+\frac{1}{30}+\frac{1}{50}+\frac{1}{75}\right)\text{MB/S}=0.2\text{MB/S}$$

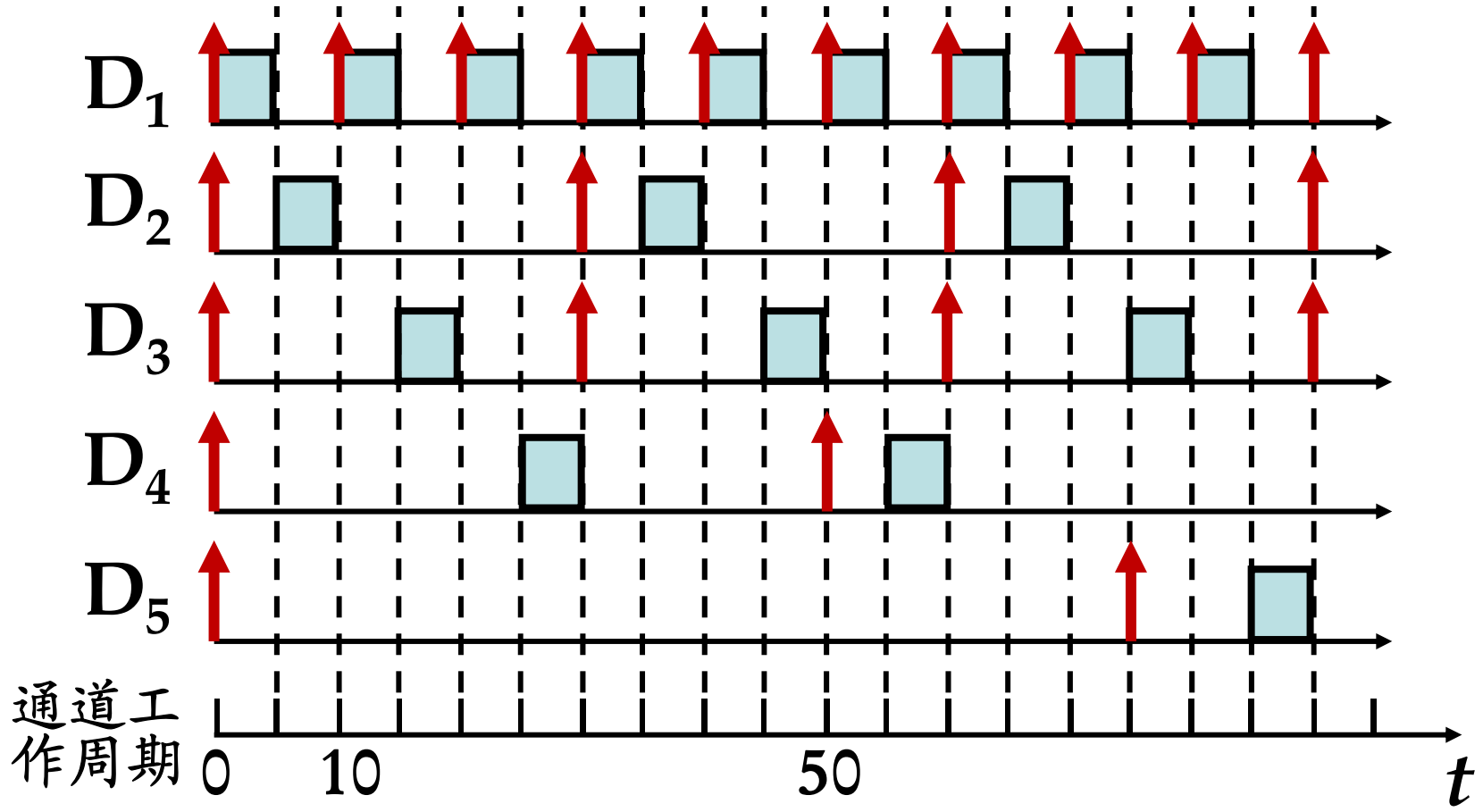
  - (2)通道流量＝实际流量＝0.2MB/s

    此时通道的工作周期为：

$$t=1/f_{BYTE}=5\mu s$$

3.4.4 通道流量的分析

- 例2 解：



3.4.4 通道流量的分析

- 例2 解：
  - (2)处理完各台设备第一次数据传送请求的时刻：

    D1：第5μs；　D2：第10μs；

    D3：第20μs；　D4：第30μs；

    D5：??? （未及时处理，丢失）

3.4.4 通道流量的分析

- 例2 解：
  - (3)发现的问题

    D5：未得到及时处理，数据丢失。

    解决方法（1/2）：
    - 1. 增加通道的最大流量
    - 2. 动态改变设备的优先级。

      例如，在30μs至70μs之间临时提高设备D5的优先级。

3.4.4 通道流量的分析

- 例2 解：
  - 解决方法（2/2）：
    - 3. 增加缓冲存储器（锁存器）。特别是对优先级比较低的设备。

      例如，只要为设备D5增加一个数据缓冲寄存器，它的第一次请求可以在第85μs处得到响应，第二次请求可以在第145μs处得到响应

3.5 外围处理机

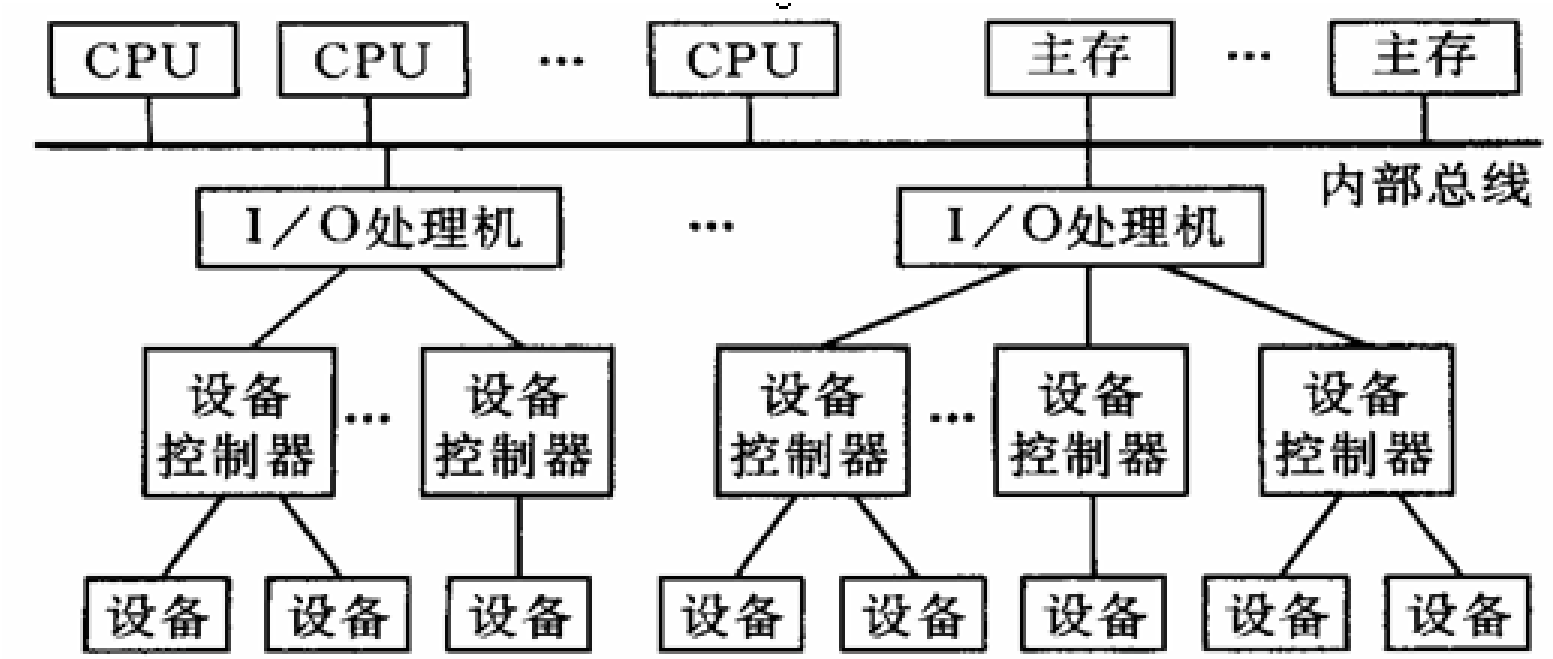
■ 通道处理机的局限性

- 通道处理机不能被看成是一台独立的处理机，因为：
  - 其指令功能简单
  - 只有面向外设控制和数据传送的功能
  - 没有大容量存贮器
- 在输入输出过程中还需要CPU处理。例如：
  - I/O的前处理和后处理
  - 错误、异常的处理等

3.5 外围处理机

- 为此，设置专门的处理机（即外围处理机，PPU）完成I/O处理，让CPU进一步摆脱对输入输出操作的控制。
- 外围处理机接近于一般的处理机，有时干脆就采用通用机，因此具有丰富的指令和较强的功能，有利于简化设备控制器，进一步承担诊断、维修、系统工作状况显示、改善人机界面等功能。

3.5 外围处理机


CPU
CPU
CPU
主存
主存
内部总线
I／O处理机
I／O处理机
设备控制器
设备控制器
设备控制器
设备控制器
设备控制器
设备
设备
设备
设备
设备
设备
设备
设备

3.5 外围处理机

- 工作原理
  - 外围处理机基本上以独立于主机的异步方式工作。
  - 外围处理机可以与主机共享主存。此时外围处理机存贮器容量较小，所执行的例行程序一般放在主存中，为各台PPU共享。
  - 外围处理机也可以不与主机共享主存。此时各外围处理机具有更强的独立性，但需要大容量内存。

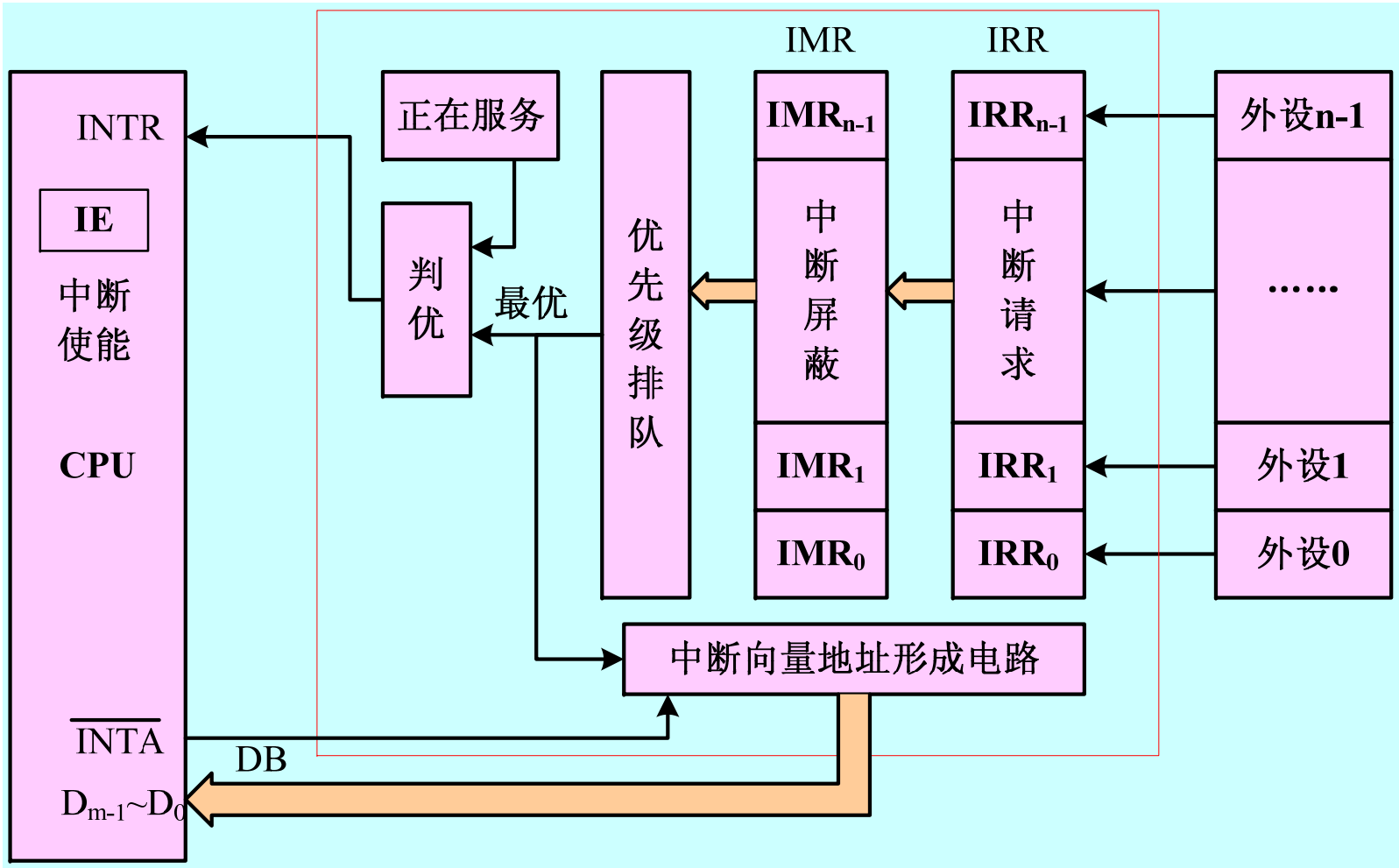
3.5 外围处理机

- 优点
  - 灵活性好。可以选择任意通道和I/O设备进行通信，主存、PPU、通道和设备相互独立，可视需要用程序动态地控制链接；
  - 大大减轻了CPU负担，提高了整个计算机系统的工作效率。因为PPU是独立的处理机，可以承担外围运算和操作控制等任务

3.5 外围处理机

- 发展
  - 让外围处理机完成更多的任务，发展出了前端机（如网络系统中的前端处理机，远程终端控制前端机等）和后台机（如数据库机）

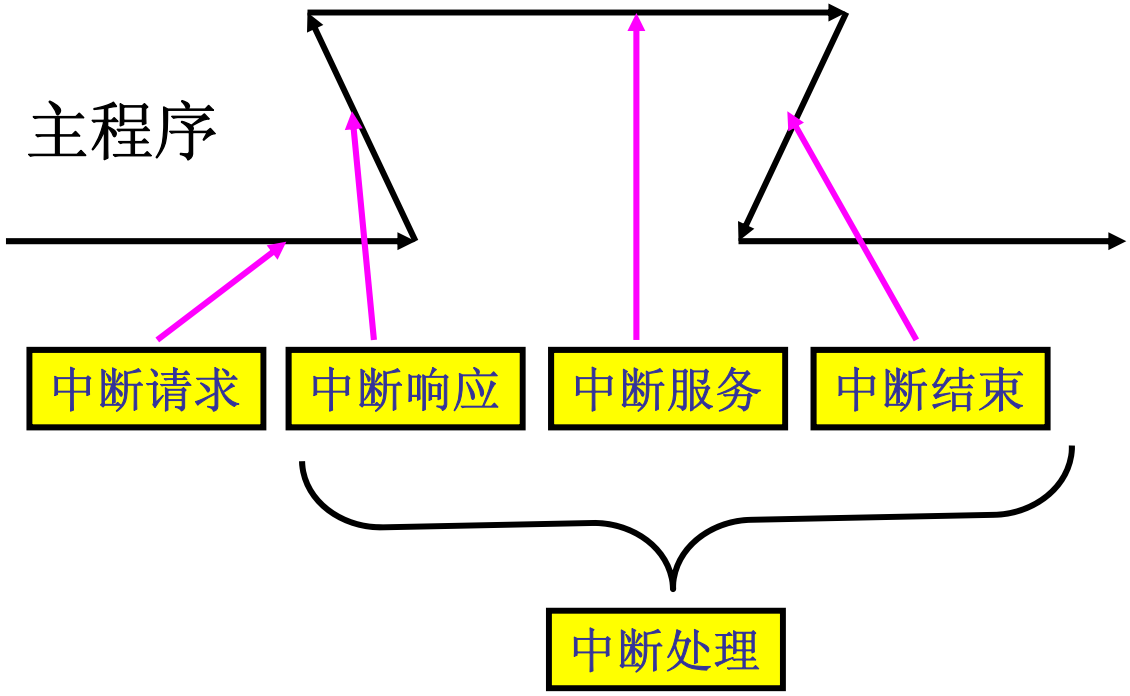
3.6 中断系统



中断系统框图

3.6 中断系统

- 中断过程



- 1.中断请求
  - （1）CPU对中断请求的监测：CPU在每条指令执行完毕后，检测CPU的中断请求引脚是否有效

3.6 中断系统

1.中断请求（续）

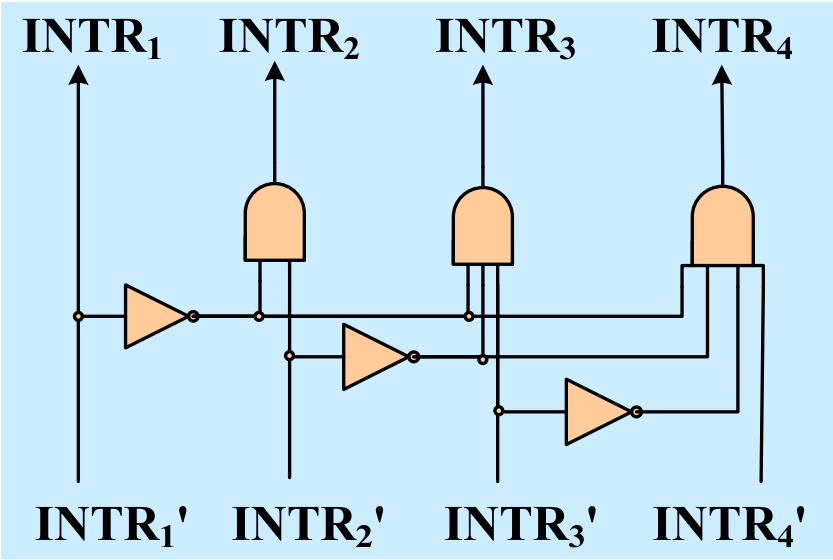
- （2）中断源（外设或其他突发事件）产生中断的方法：置位接口中的**中断请求触发器**；系统中所有的中断源的中断请求触发器构成一个**中断请求寄存器**。
- （3）CPU对中断请求的屏蔽：指CPU是否响应某个中断源的中断请求；由接口中的**中断屏蔽触发器**控制：0--开放， 1--屏蔽；所有中断源的中断屏蔽触发器构成一个**中断屏蔽寄存器**，CPU可以对其读写。

3.6 中断系统

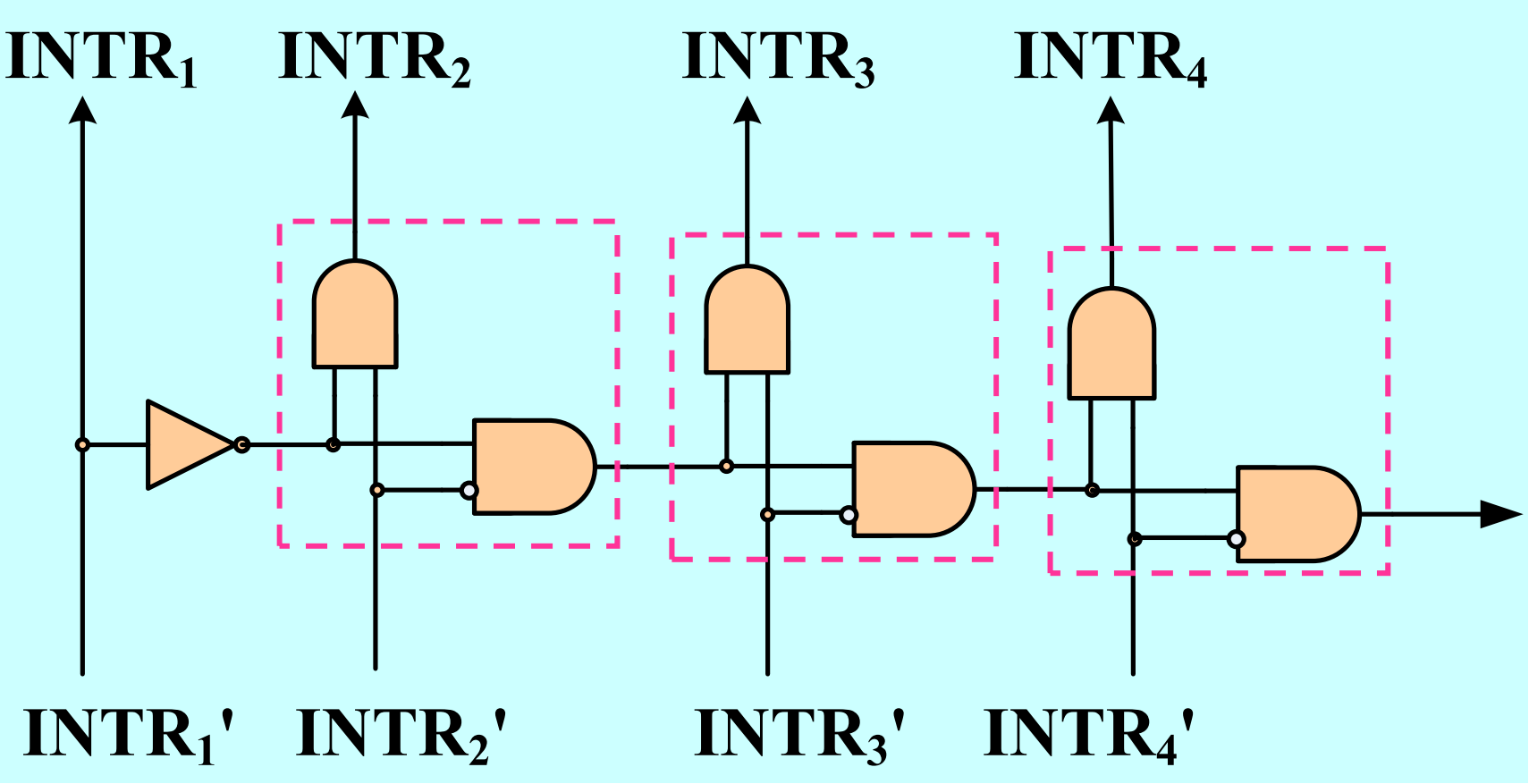
1.中断请求（续）

- （4）中断请求的优先级判定：指当多个中断源同时发生中断请求时，需要根据优先级排队；通常采用硬件排队电路或者软件判优

硬件排队电路1
（集中式）

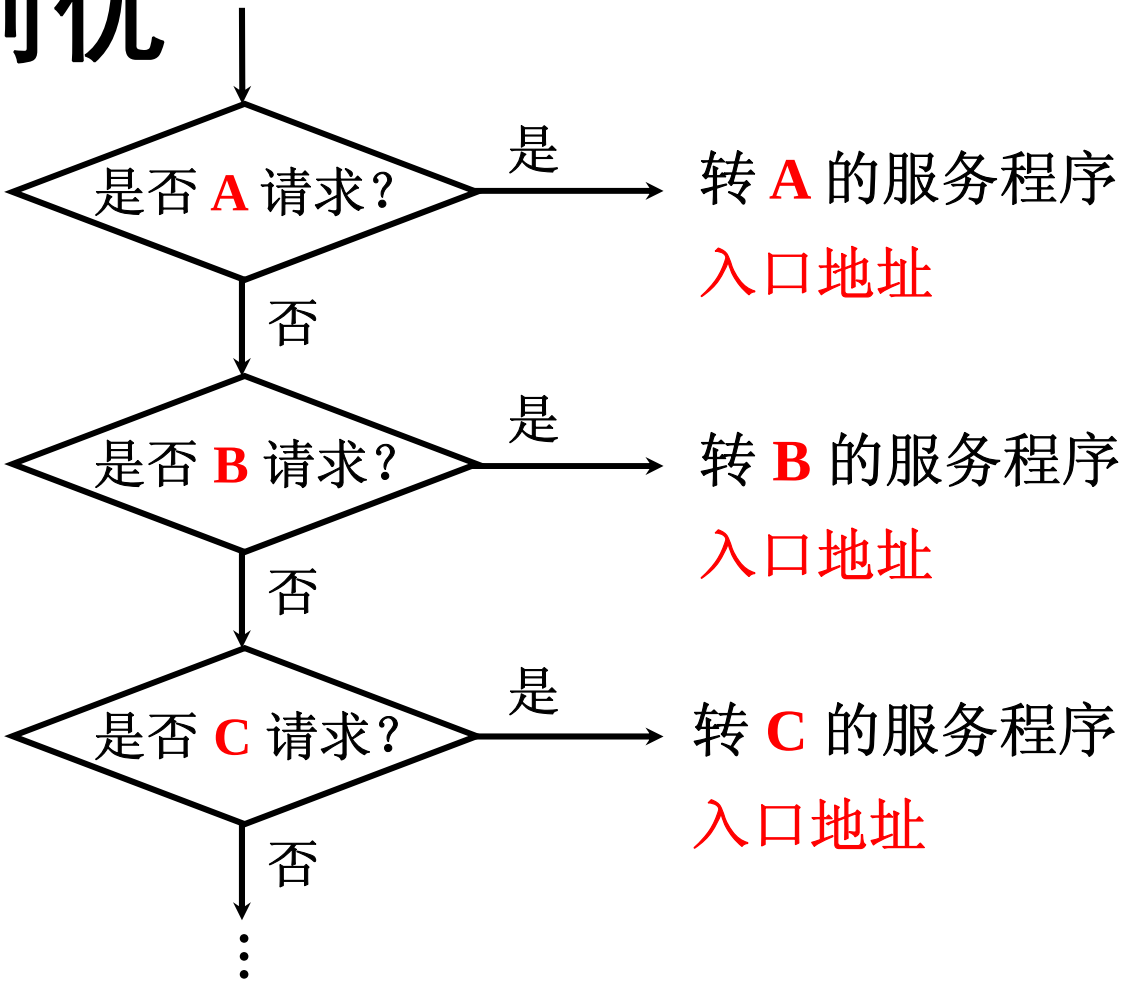


硬件排队电路2（链式、分散式）



在排队链中，越靠前的优先级越高

软件判优



- 中断源优先级的优先级取决于被询问的顺序，最先被询问的，优先级最高。

3.6 中断系统

2. 中断响应

- （1）中断响应时间：当前指令执行完
- （2）中断响应条件：CPU的INTR引脚有效（有中断请求），CPU处于开中断状态
- 具体到某个中断源，中断响应条件为：
  - 外设有中断请求（IRRi=1）
  - 该中断请求没有被CPU屏蔽（IMRi=0）
  - 该中断请求优先级最高
  - 该中断请求优先级高于正在被服务的中断源
  - 中断使能有效（IE=1）

3.6 中断系统

2. 中断响应

- （3）中断响应动作：执行中断隐指令。
  - 关中断（IE=0）
  - 保存程序断点（PC压栈，标志寄存器压栈）
  - 转入中断服务程序执行（中断服务程序入口地址→PC）
- 中断隐指令由硬件来实现，对程序员是透明的。

3.6 中断系统

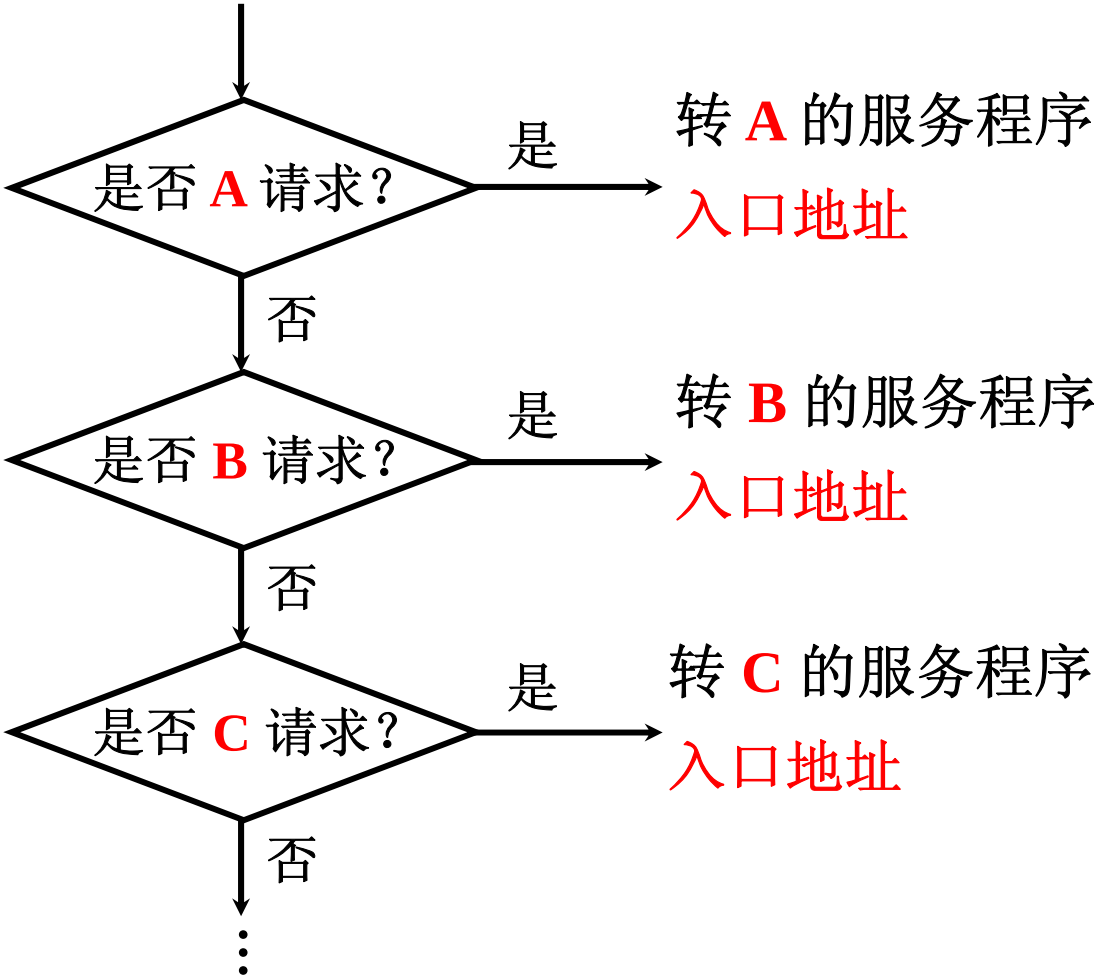
2. 中断响应

- （4）中断源的识别：如何找到中断服务程序入口地址？
  - ①向量中断：由外设接口硬件产生后送给CPU
- 中断向量：中断源的中断服务程序入口地址
- 中断向量表（IVT）：所有中断源的中断向量按顺序存放在内存，形成一张表
- 中断向量地址（指针）：指向IVT的地址
- CPU通过INTA#信号读取中断向量地址，然后启动一个中断响应周期从内存读取中断向量，置入PC

3.6 中断系统

■②软件查询：

- 中断隐指令转入一个总中断服务程序，在此逐个查询各个中断源有否中断。
- 查询的顺序决定了中断源的优先级。

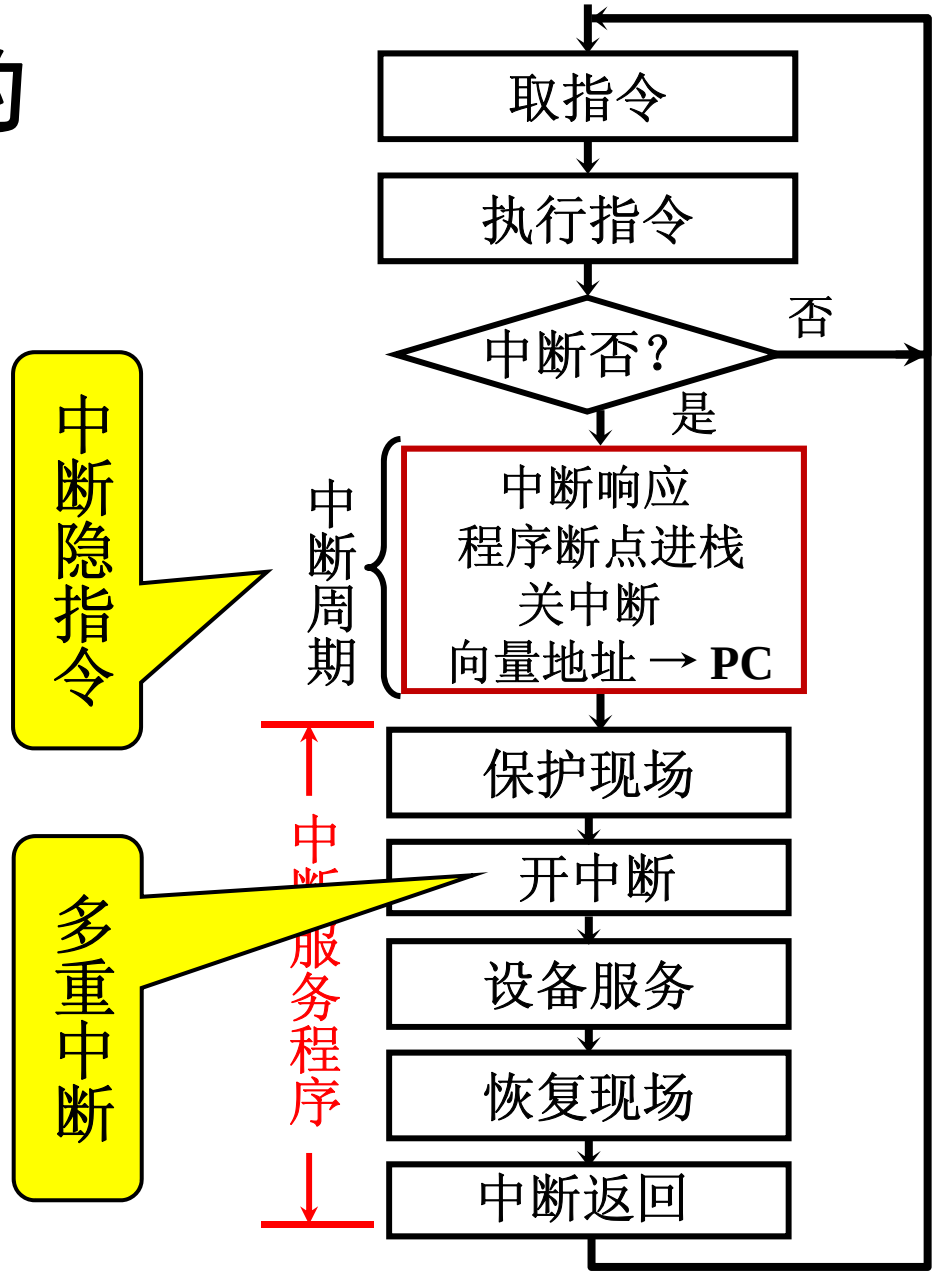


3.6 中断系统

2. 中断响应

- （5）多重中断的概念：中断嵌套。
  - 单重中断：不允许中断 现行的 中断服务程序
  - 多重中断：允许级别更高 的中断源中断 现行的 中断服务程序
- 实现中断嵌套的条件：
  - 软件：在中断服务程序中要“开中断”
  - 硬件：高优先级的中断源请求可以送达CPU
- 与中断屏蔽不同

具有中断功能的CPU运行流程



3.6 中断系统

- **3. 中断服务：** 执行中断服务程序
  - 中断服务程序处理流程：
    - **保护现场：** 保存相关寄存器内容（压栈）
    - **中断服务：** 进行IO传送操作或突发事件处理
    - **恢复现场：** 恢复相关寄存器内容（出栈）
    - **中断返回：** 程序断点出栈→PC
- **4. 中断返回：** 执行中断服务程序中的IRET指令（标志寄存器出栈，PC出栈）

本章重点

- 输入输出系统
- 总线的控制方式
- 通道工作原理
- 通道类型
- 通道流量的分析

第3章 作业3