东南大学计算机学院

计算机系统组成

主讲教师： 徐造林

第9章 输入输出系统

- 输入输出组织：控制外设与内存或CPU之间进行数据交换的机构；
- 把I/O设备及其接口线路、控制部件、通道或I/O处理器以及I/O软件统称为输入输出系统。
- 本章主要介绍I/O接口的功能和结构、I/O设备的编址和寻址、主机和外设间进行数据传送的各种输入输出控制方式。

9.1 I/O接口

- 接口定义
  - 定义：是CPU与“外部世界”的连接电路，负责“中转”各种信息。
  - 分类：存储器接口和I/O接口。
  - 位置：介于系统总线与外部设备之间。
- 完成各个外设和主机之间的同步与协调、工作速度的匹配和数据格式转换的逻辑部件称为I/O接口。
- 计算机中各种I/O控制器（I/O处理器）或设备控制器都是I/O接口。

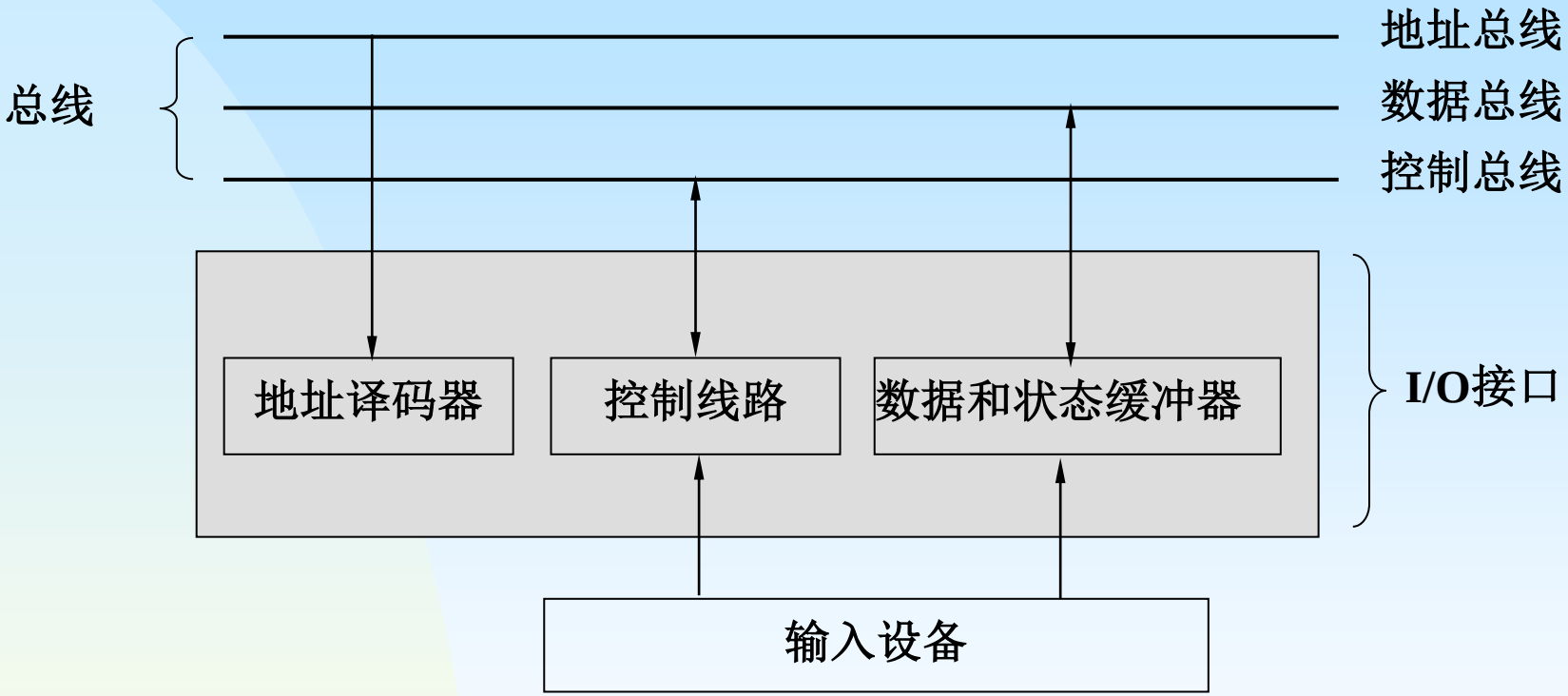


图9.1 用于输入设备的I/O接口

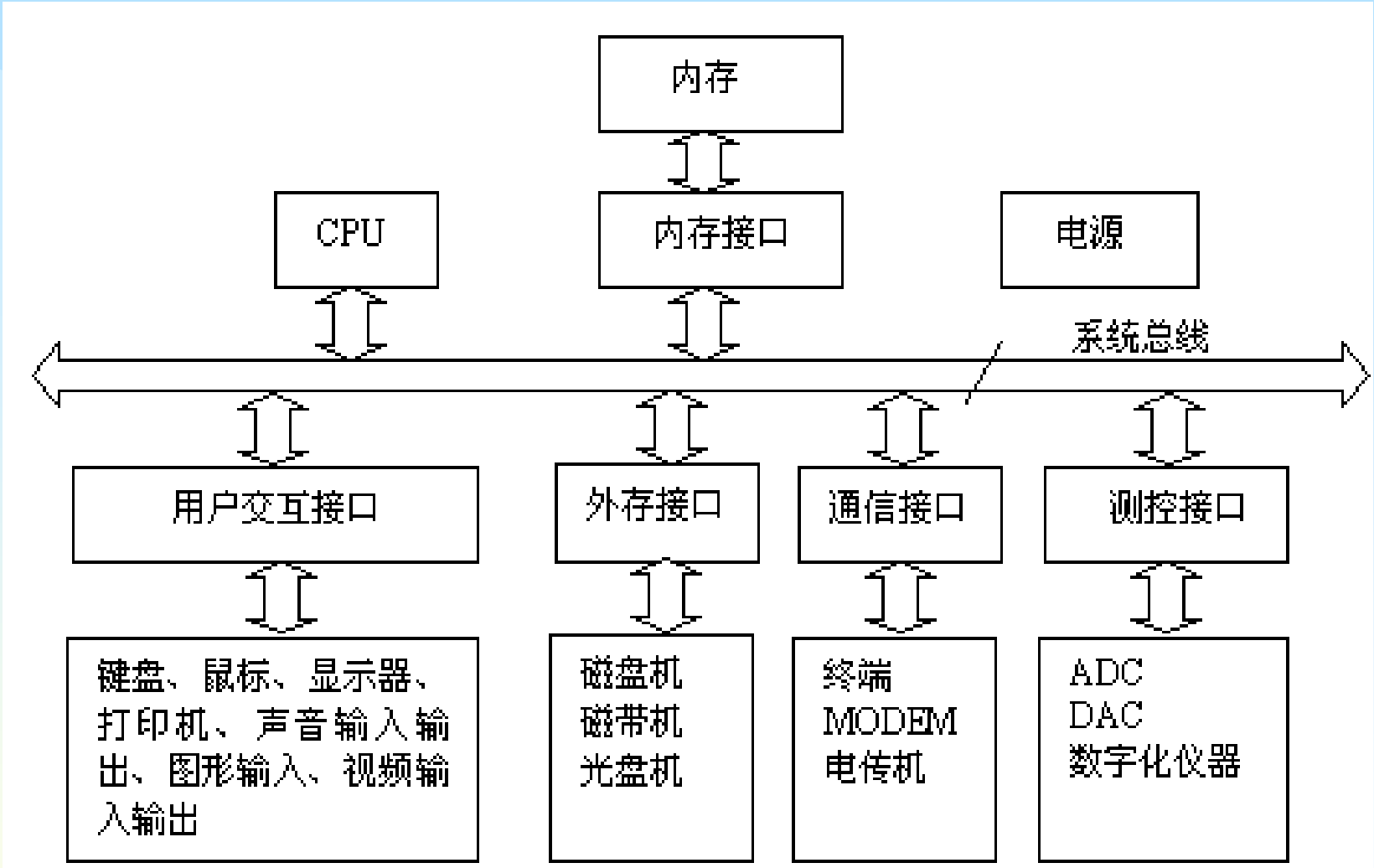


图9.2 计算机I/O接口类型

I/O接口与I/O设备

- 不同I/O设备对应I/O接口不同；
- I/O接口受CPU控制，I/O设备受I/O接口控制；
- 为增加通用性，I/O接口的接口电路一般均具有可编程功能；
- 微机的应用离不开与外部设备接口的设计、选用和连接。

9.1.1 I/O接口的功能

（1）数据缓冲：解决主机和外设工作速度的匹配问题；

（2）错误或状态检测：提供状态寄存器供CPU查用；

（3）控制和定时：接收系统总线来的控制和定时信号；协调内部资源与外设间动作的先后关系，控制数据通信过程；

（4）数据格式转换；

（5）与主机和设备通信：必须通过I/O接口完成主机与设备之间的通信。

a）I/O接口与主机侧进行通信：

- 进行地址译码，以确定是否选中本设备；
- 接收控制信息，确定数据传送的方向等；
- 接、送数据或状态信息。

b）I/O接口与设备进行通信：

- 将控制寄存器中的命令译码，输出到外部接口的控制线上；
- 发送数据缓冲寄存器的数据到外部接口的数据线上；
- 接收外设的状态或数据信息，送到接口中的状态寄存器或数据缓冲寄存器中。

分析和设计I/O接口的方法

（1）分析接口两侧的情况

- 内部接口通过系统总线与内存、CPU相连；
- 外部接口通过各种接口电缆将其连到外设上。

（2）进行信号转换

- 数据信号转换和控制信号转换。

（3）合理选用接口芯片

（4）接口驱动程序的分析和设计

9.1.2 I/O接口的结构

- 接口硬件组成

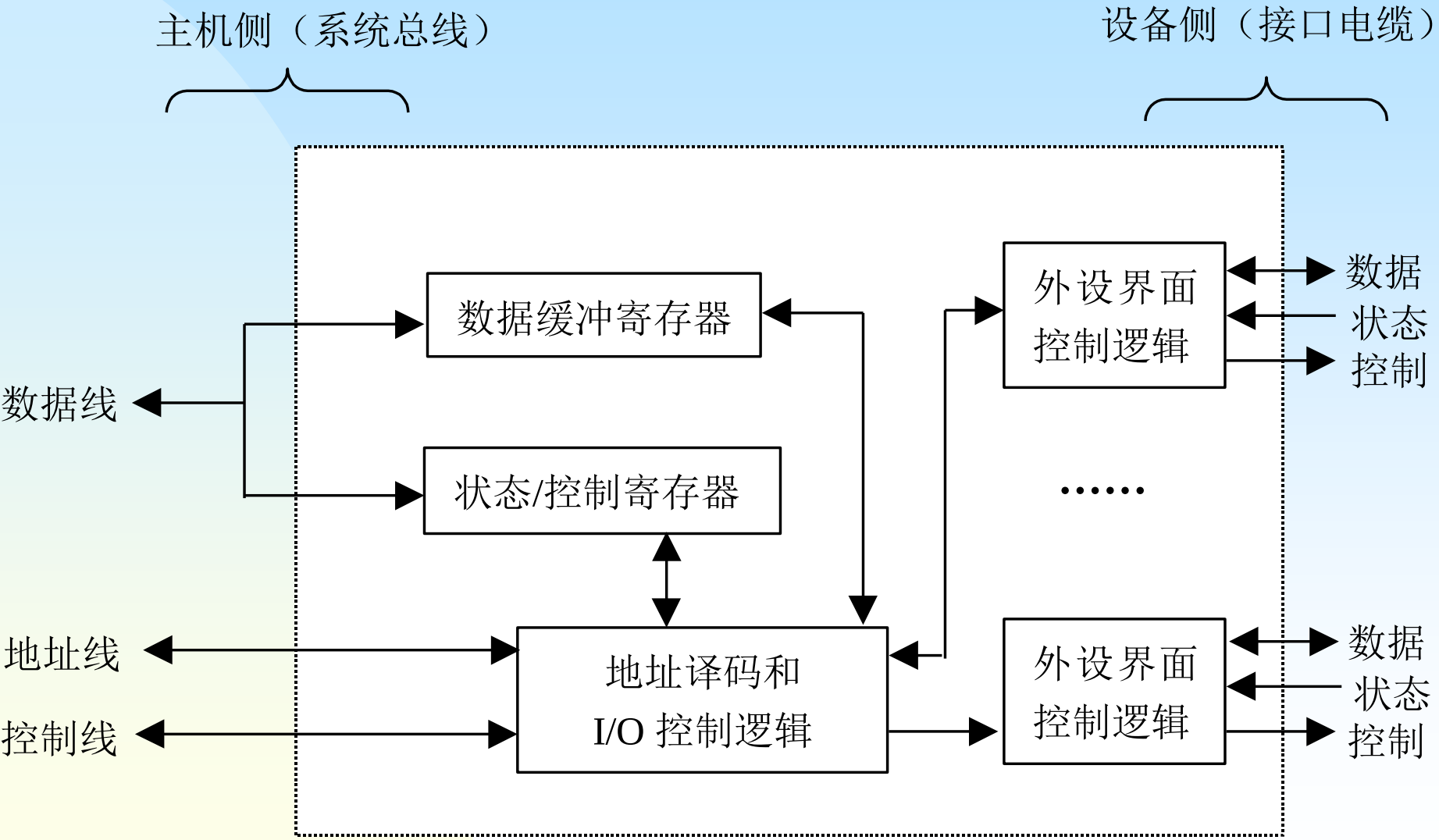


图9.3　I/O接口的通用结构

- 接口软件（设备驱动程序）
  - 初始化程序段：设置接口工作方式及初始条件。
  - 传送方式处理程序段：CPU针对不同的I/O设备有不同的处理方式。
  - 主控程序段：完成接口任务的程序。
  - 程序终止与退出程序段：接口电路硬件保护及操作系统中数据恢复。
  - 辅助程序段：提供人-机对话手段。

9.1.3 I/O接口的分类

(1) 按数据传送方式分，有并行接口和串行接口

- 主机侧的内部接口，进行并行传输；
- 外设侧的外部接口，有串行和并行两种传送方式。

(2) 可编程接口和不可编程接口

(3) 按通用性来分，有通用接口和专用接口

- 通用接口可供多种外设使用，如Intel 8255、Intel 8212；
- 专用接口是为某类外设或某种用途专门设计的，如Intel 8279可编程键盘/显示器接口、Intel 8275可编程CRT控制器接口。

(4) 按数据传送的控制方式来分，有程序控制方式接口、程序中断方式接口和直接内存访问方式接口。

(5) 按设备的连接方式来分，有点对点接口和多点接口。
点对点接口：如打印机、键盘、调制解调器等设备；
多点接口：SCSI接口和P1394接口等。

9.1.4 I/O端口的寻址方式

- I/O端口寻址：让CPU能方便地找到要进行信息交换的设备；

  - I/O端口：CPU与I/O设备直接通信的地址。

操作系统在I/O中的作用

- 使用户程序通过一些简单的命令或系统调用就能使用各种I/O设备。
- I/O软件的四个层次
  - 用户层I/O软件；
  - 与设备无关的操作系统I/O软件；
  - 设备驱动程序；
  - I/O中断处理程序。

I/O端口的编址方式

(1) 独立编址方式：对所有的I/O端口单独进行编号，成为一个独立的I/O地址空间。

- 需要用专门的输入输出指令来访问I/O端口。

(2) 统一编址方式：将主存地址空间分出一部分地址给I/O端口进行编号。

- 访存指令和输入输出指令相同。

- 有关主存的寻址方式都可用于I/O端口的寻址。外设或I/O寄存器数目几乎不受限制。
- 主存空间减少；址线都需参与地址译码，使译码电路变复杂。

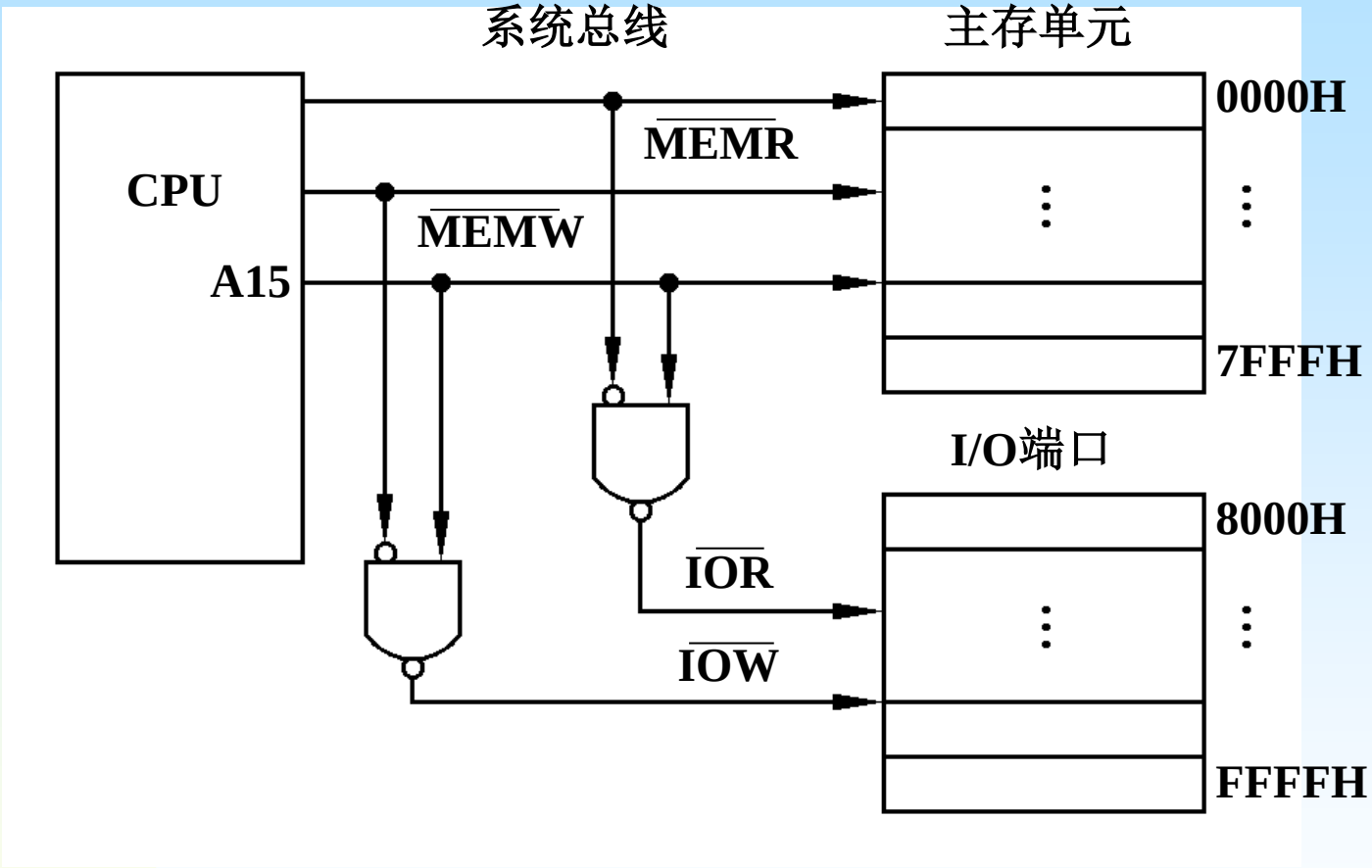


图9.4 统一编址方式

- 寻址速度快；专用I/O指令，使得程序清晰。
- 程序设计灵活性差些；控制逻辑较复杂。

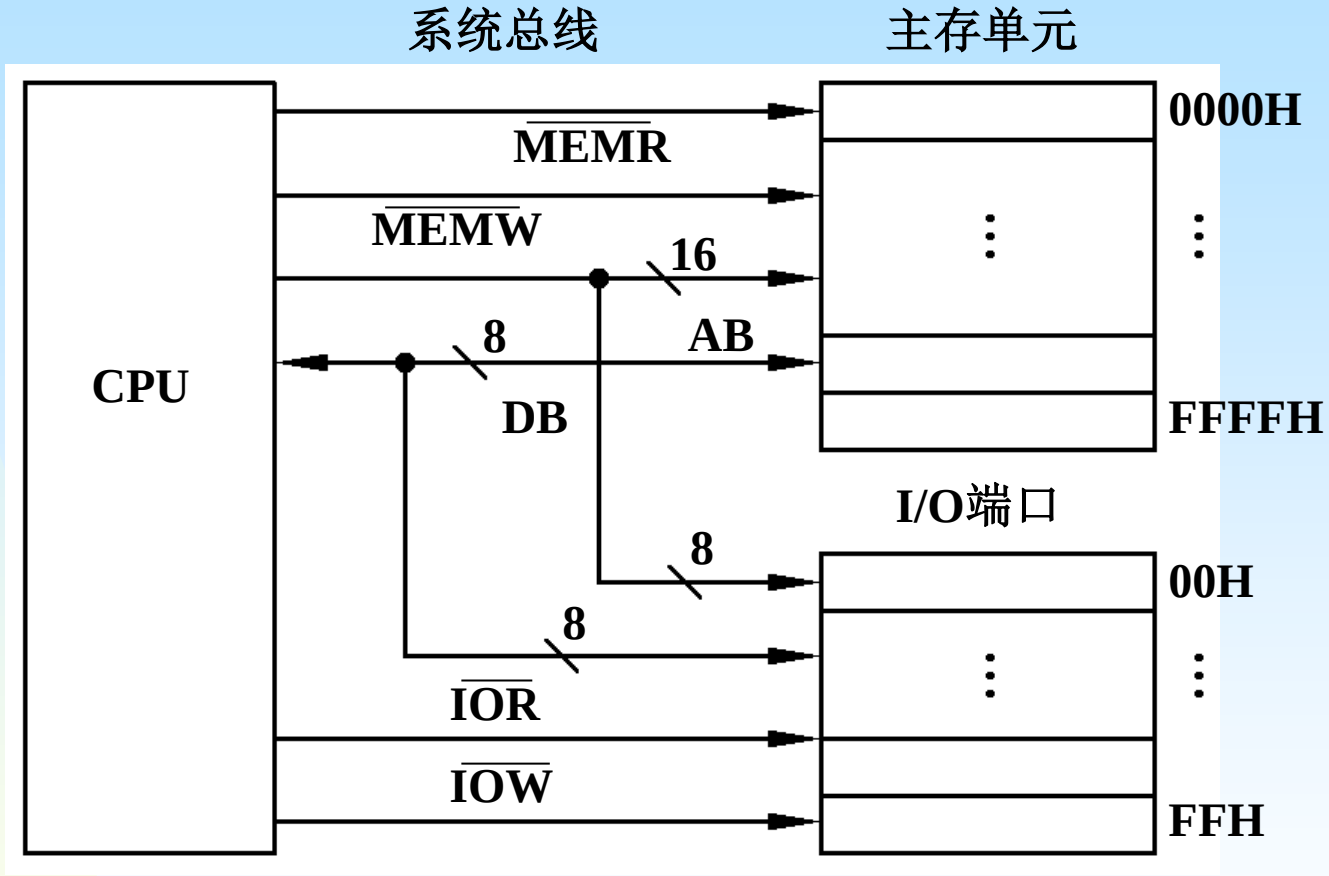


图9.5 独立编址方案

- 独立编址方式的端口访问
  - 8088/8086采用I/O端口与累加器之间的传送

```
IN      AX，PORT  ；直接寻址
IN      AL，PORT
OUT     PORT，AX
OUT     PORT，AL

MOV     DX，PORT  ；寄存器间接寻址
IN      AX，DX
IN      AL，DX
OUT     DX，AX
OUT     DX，AL
```

- **I/O接口芯片片选译码（集中译码）**
  - **74LS138在PC/XT机系统板中芯片的译码**

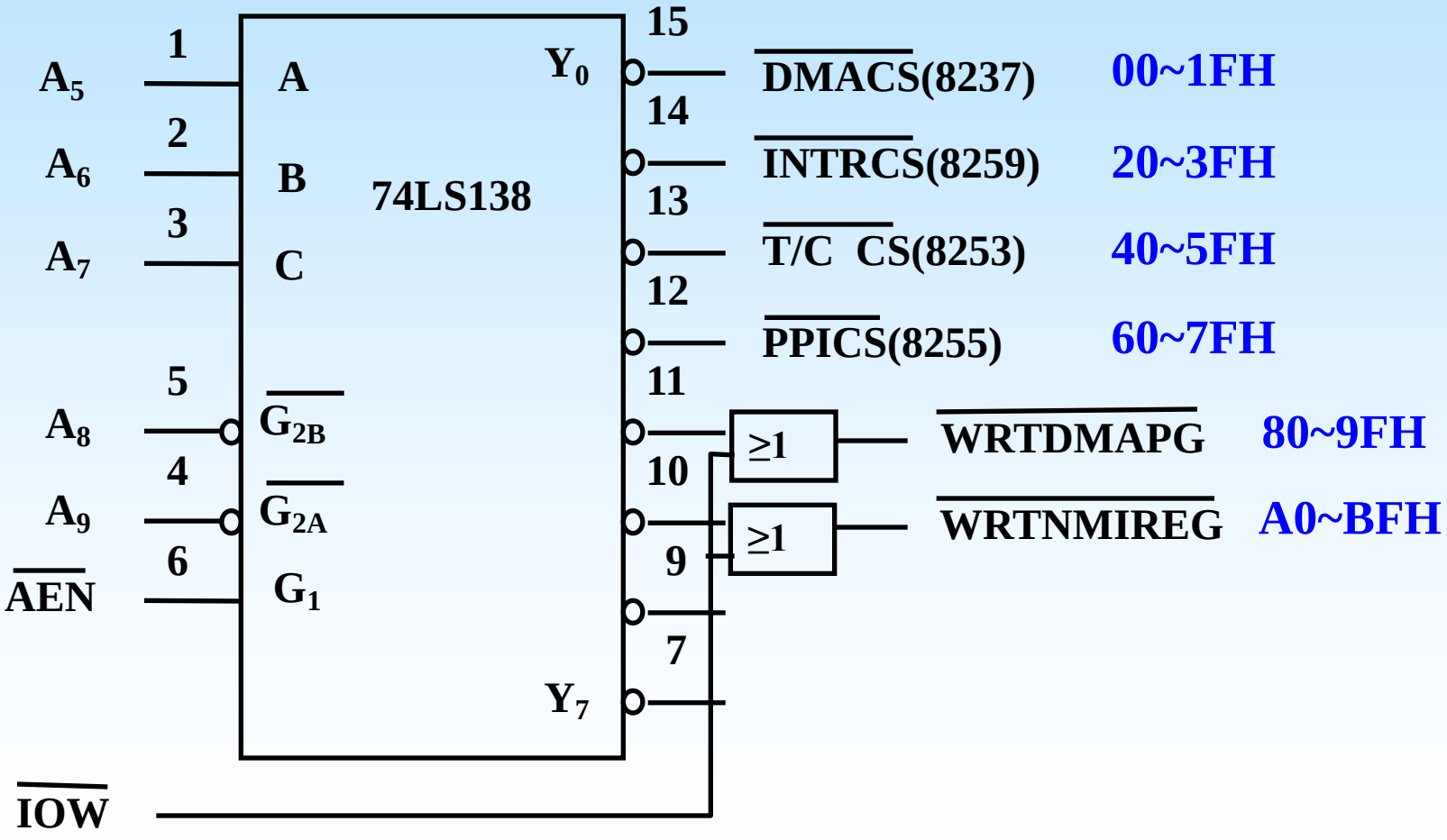


表9.1 部分外设控制器的I/O地址分配表

输入/出设备	I/O地址	占用地址数
DMA控制器1	000-01FH	32
中断控制器1	020-03FH	32
定时器/计数器	040-05FH	32
键盘控制器	060-06FH	32
实时时钟，		
NMI屏蔽寄存器	070-07FH	16
DMA页面寄存器	080-09FH	32
中断控制器2	0A0-0BFH	32
DMA控制器2	0C0-0DFH	32
硬盘控制器2	170-177H	8
硬盘控制器1	1F0-1F8H	8

表9.1续

输入/出设备	I/O地址	占用地址数
游戏I/O口	200-207H	8
并行打印机口2	278-27FH	8
串行口4	2E8-2EFH	8
串行口2	2F8-2FFH	8
软盘控制器2	370-377H	8
并行打印机口1	378-37FH	8
单色显示器/打印适配器	3B0-3BFH	16
彩色/图形监视器适配器	3D0-3DFH	16
串行口3	3E8-3EFH	8
软盘控制器1	3F0-3F7H	8
串行口1	3F8-3FFH	8

9.2 I/O数据传送控制方式

9.2.1 I/O控制方式类型

1. 程序直接控制方式（查询方式）

- 从I/O接口取得外设和接口的状态，根据状态来控制外设和主机的信息交换。

2. 程序中断控制方式

- 执行相应的I/O指令，将启动命令发送给相应的I/O接口和外设，然后CPU继续执行其他程序。

3. 直接存储器存取方式

- 简称为DMA方式，用于高速设备和主机的数据传送，采用成批数据交换方式。
- 用专门的硬件（DMA控制器）来控制总线进行数据交换。

4. 通道和I/O处理器方式

- 获得CPU和外设之间更高的并行性，让种类繁多、物理特性各异的外设能以标准的接口连接到系统中。

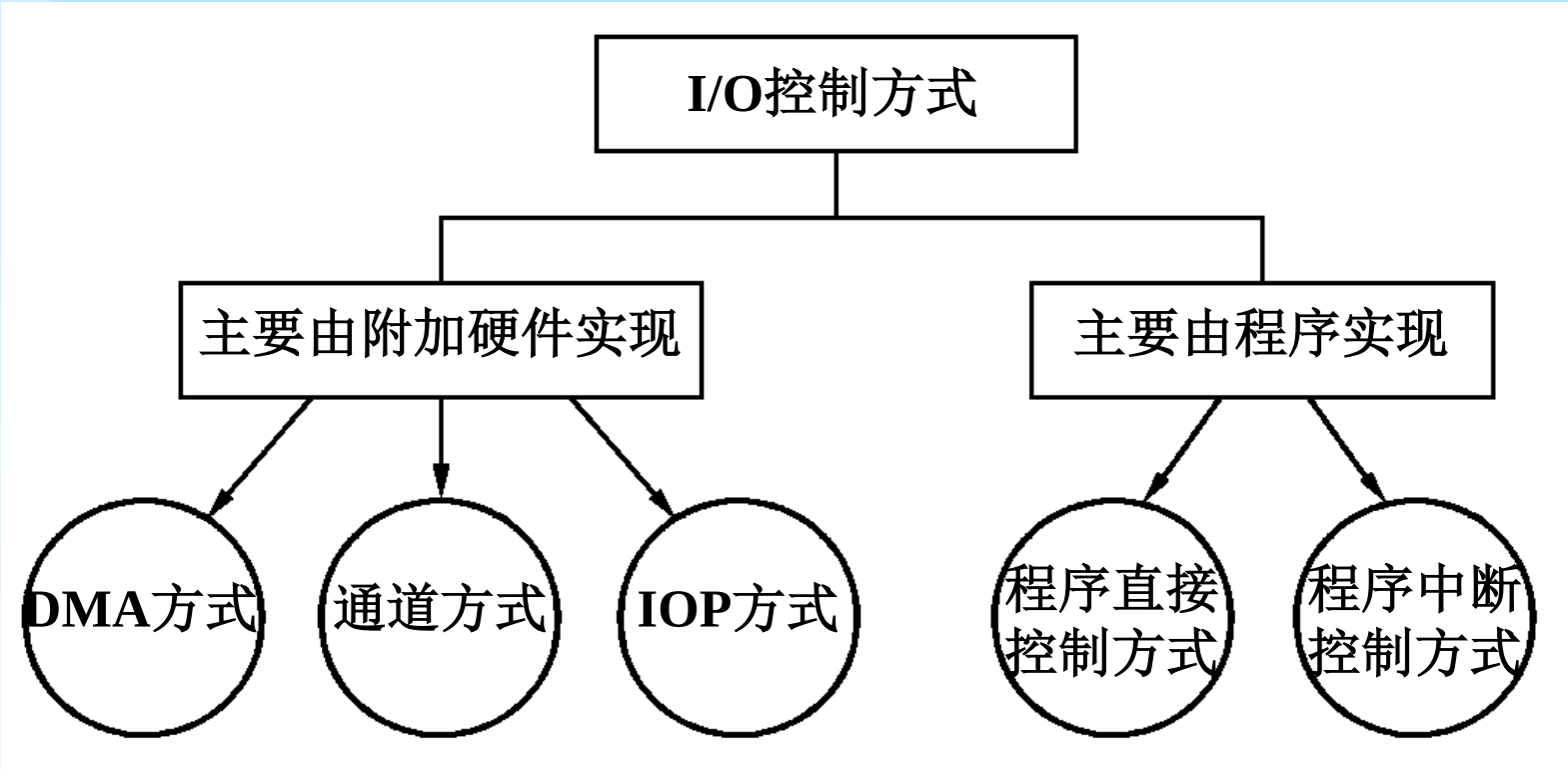
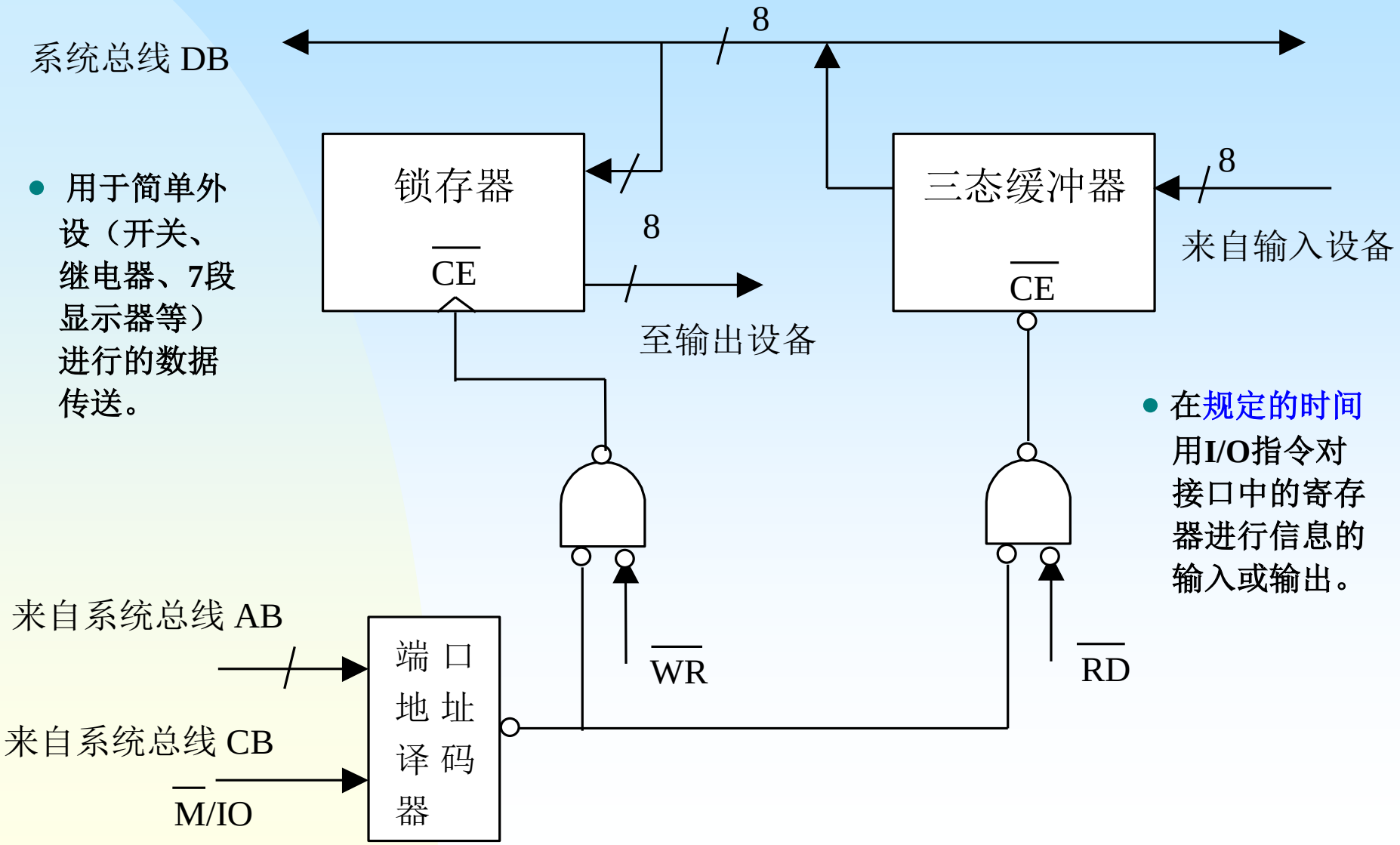


图9.6　外部设备的I/O控制方式

9.2.2 程序直接控制方式

1. 无条件传送方式（同步传送方式）

- 用于简单外设（开关、继电器、7段显示器等）进行的数据传送。



- 在规定的时间用I/O指令对接口中的寄存器进行信息的输入或输出。

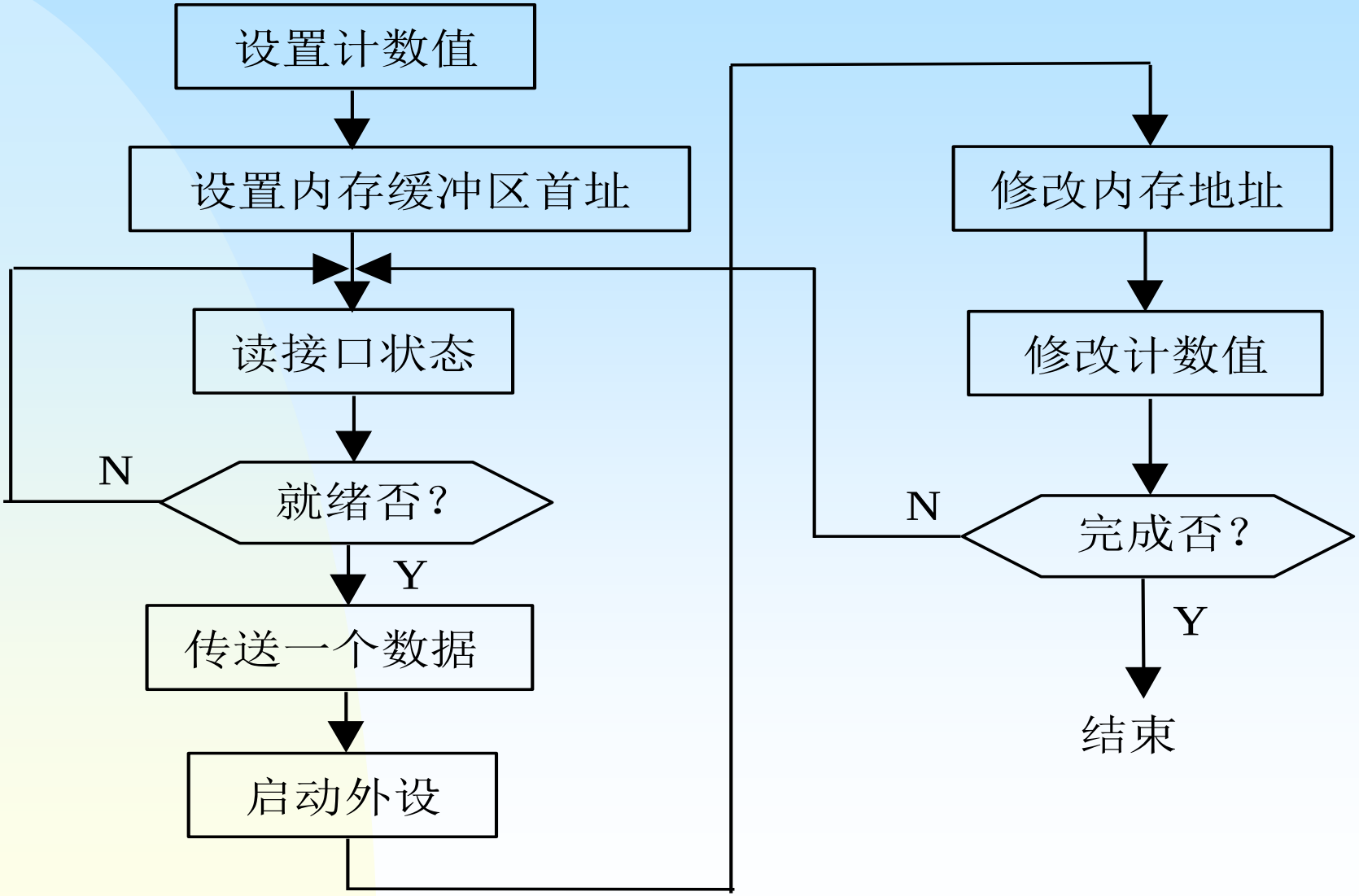
2. 条件传送方式（异步传送方式）

- 通过程序查询取得外设和接口的状态（就绪、忙、完成），根据这些状态来控制外设和主机的信息交换。

▲ 程序查询方式的特点

- 程序查询方式简单、易控制、外围接口控制逻辑少；
- CPU与外设完全串行工作，效率低、速度慢；
- CPU会浪费许多处理器时间。

▲ 程序查询方式数据传送流程



- 从键盘读取一行字符，存储在内存缓冲区，并在显示器上回显的程序

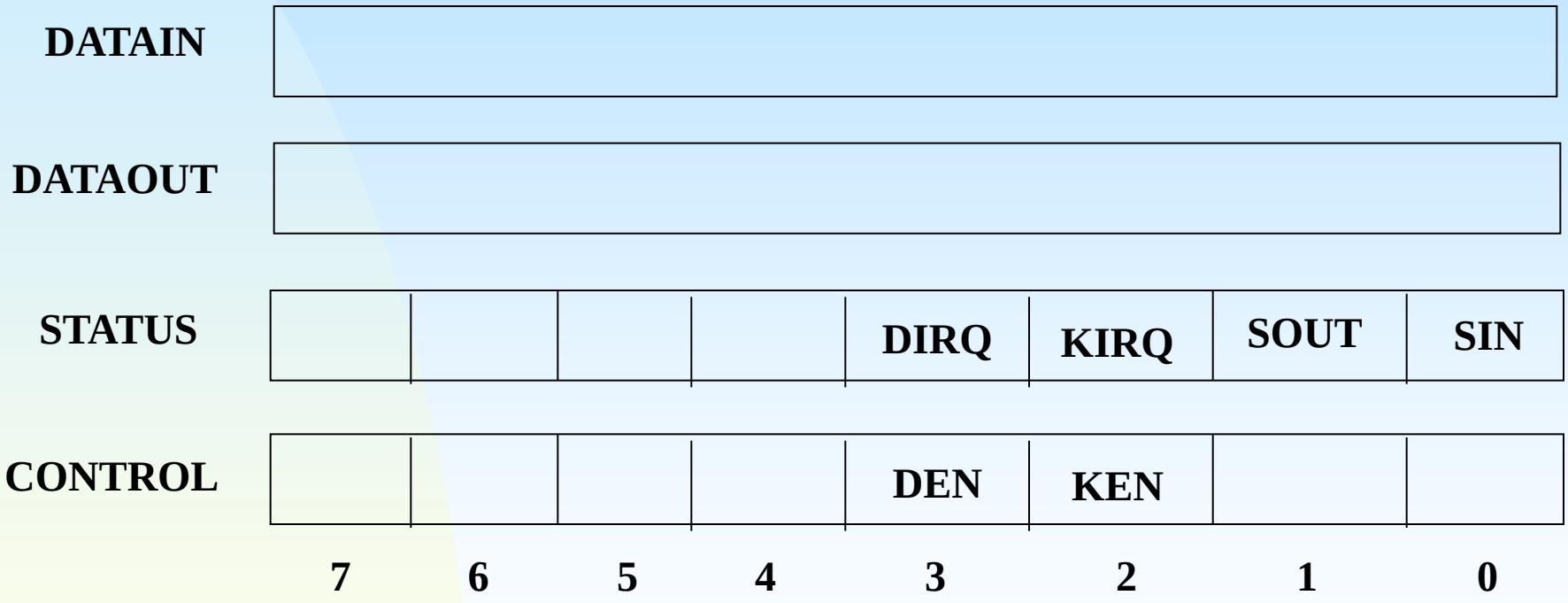


图9.7　键盘和显示器接口中的寄存器

```
        Move        #LINE，R0          初始化存储缓冲区指针
WAITK   TestBit     #0，STATUS         测试SIN
        Branch=0    WAITK              等待键盘输入
        Move        DATAIN，R1         读字符
WAITD   TestBit     #1，STATUS         测试SOUT
        Branch=0    WAITD              等待显示器准备好
        Move        R1，DATAOUT        将字符送显示
        Move        R1，（R0）+        存储字符并将指针加1
        Compare     #0DH，R1           检查是否回车符？
        Branch≠0    WAITK              不是，则取下一字符
        Move        #0AH，DATAOUT      否则，输出换行
        Call        PROCESS            调用一个子程序处理输入的字符行
```

9.3 程序中断方式

9.3.1 中断的概念

- 中断：
  - 由于内部/外部事件或由程序的预先安排引起CPU中断正在执行的程序，转到相应的服务程序中去。
  - 中断源：能够引发CPU中断的来源。

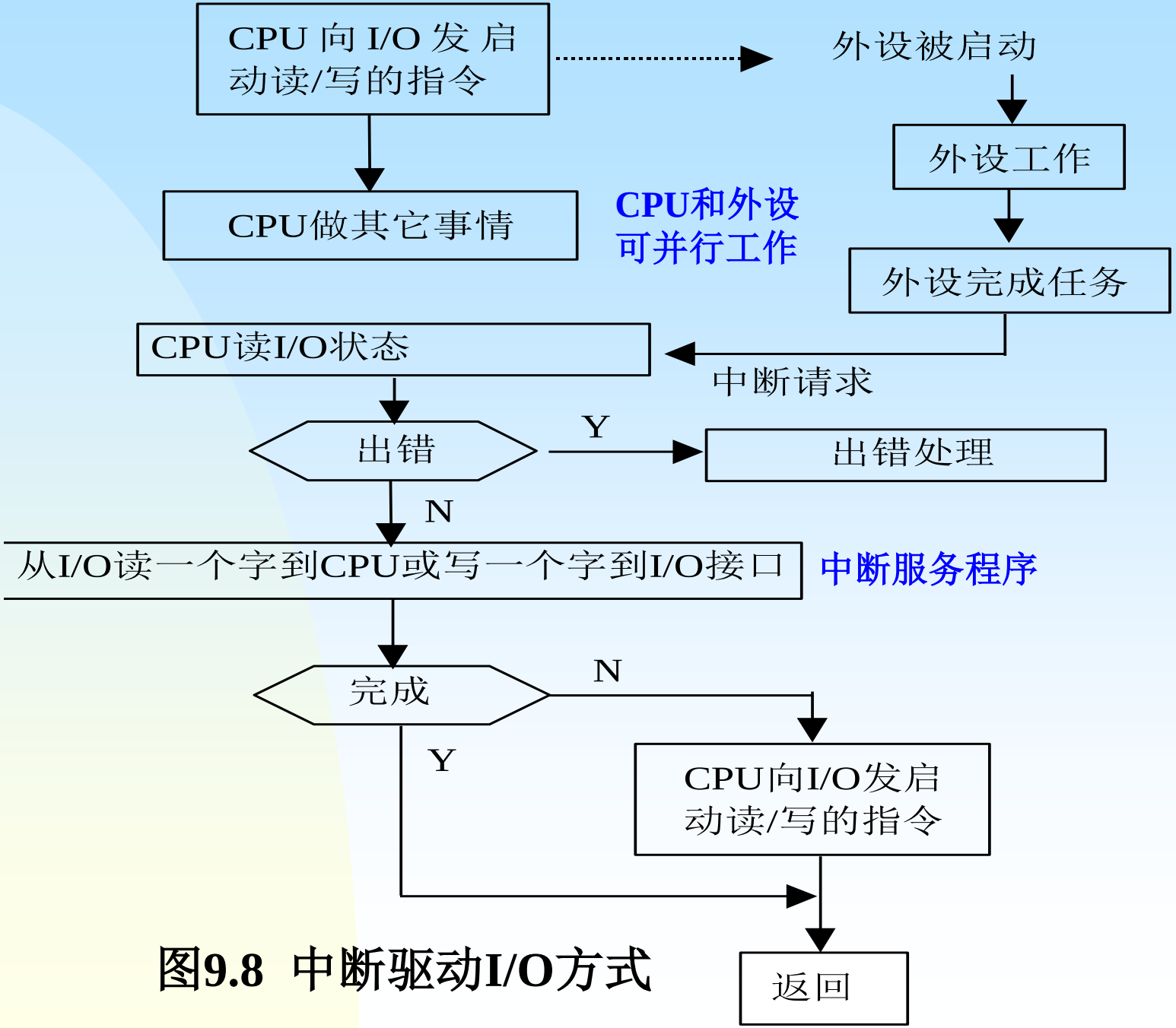


图9.8 中断驱动I/O方式

- 中断I/O方式特点：
  - 充分发挥CPU的高速处理能力。
  - 实现外设与CPU 的并行

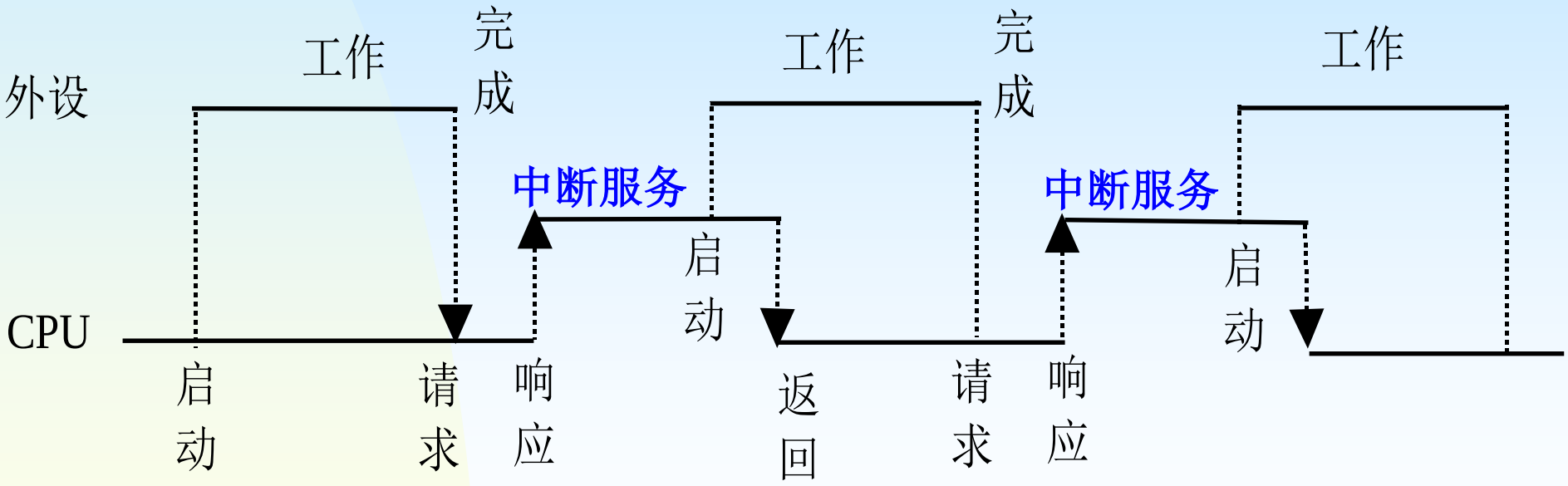
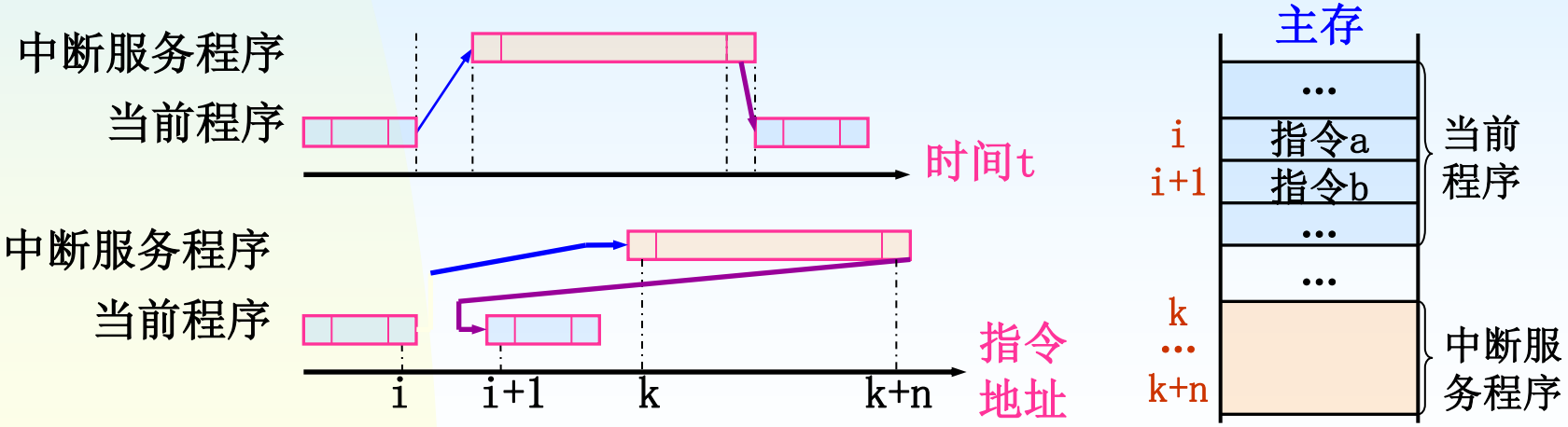


图9.9　CPU与外设并行工作

- 现代计算机系统都配有完善的中断系统
  - 中断系统是计算机实现中断功能的软、硬件的总称。
  - **CPU**：中断响应和处理，
  - 外设接口：中断请求和控制逻辑，
  - 操作系统：中断服务程序。
  - 中断硬连线路和中断服务程序有机结合，共同完成和控制中断过程。

中断的相关术语：

- 中断请求—表示有急待处理的突发事件的信号；
- 中断源—能够产生中断请求的部件(或接口或设备)；
- 中断服务程序—中断请求(突发事件)对应的处理程序；
- 中断响应—从当前程序转入中断服务程序的过程；
- 中断服务—执行中断请求对应中断服务程序的过程；
- 中断返回—从中断服务程序返回当前程序的过程；
- 中断处理—中断服务及中断返回的总和。



- 中断的分类

1. 内中断

- 由处理器内部的异常事件引起的中断
  - 硬故障中断：电源掉电、存储器线路错等；
  - 程序性中断：CPU执行某个指令而引起的发生在处理器内部的异常事件；
  - 如除数为0，溢出、断点、单步跟踪、访问超时、堆栈溢出、缺页、地址越界、数据格式错等。

▲ 程序性中断可分为失效、自陷和终止三类。

- 失效：在引起失效的指令启动后、执行前被检测到的一类例外事件。在中断处理程序完成后，应回到该条指令，重新启动并执行。

- 自陷：在产生自陷的指令执行完后才被报告的一类例外事件，中断处理程序完成后，回到主程序中该条指令的下一条继续执行（如，INT n指令）。

- 终止：对引起异常的指令的确切位置无法确定的一类例外事件，出现这类严重错误时，原程序无法继续执行，只好终止，而由中断服务程序重新启动操作系统。

2. 外中断

- 由外设完成任务或出现特殊情况引起（任务完成、打印机缺纸、磁盘检验错、键盘输入等）。

3. 80X86处理器的中断系统分类

- 外部中断（硬中断）：通过处理器的中断请求线INTR(可屏蔽中断)和NMI(不可屏蔽中断)来实现请求的中断。
- 内部中断（软中断）：由处理器内部产生而不通过中断请求线请求，为不可屏蔽中断。
- 中断调用

▲ 中断识别：找到哪一个中断源发出的中断请求；

- 目的：获得中断处理程序入口地址；
- 方法：向量中断和程序查询。

▲ 中断优先级：给每个中断源指定CPU响应的优先级。

▲ X86 CPU组成的微机系统中断

表9.2　微处理专用中断

中断号	名　　称	XT 型向量	AT 型向量	控　制　权
0	除零数	0237:56E8	0281:56E8	DOS-Kernel
1	单步	0070:075C	0070:075C	DOS-BIOS
2	NMI	0BA9:0016	0BF7:0016	DOS STACKS
3	断点	0070:075C	0070:075C	DOS-BIOS
4	溢出	0070:075C	0070:075C	DOS-BIOS
5	屏幕打印	F000:FF54	F000:FF54	ROM-BIOS
6	保留	F000:FF23	F000:E14F	ROM-BIOS
7	保留	F000:FF23	F000:EE6F	ROM-BIOS

表9.3　硬中断

中断号	名　　称	XT 型向量	AT 型向量	控　制　权
8	日时钟中断	0BA9:00AB	0BF7:00AB	DOS STACKS
9	键盘中断	0BA9:0125	0BF7:0125	DOS STACKS
0AH	保留/从片中断	F000:FF23	F000:EF6F	ROM-BIOS
0BH	串行口 2 中断	F000:FF23	F000:EF6F	ROM-BIOS
0CH	串行口 1 中断	F000:FF23	F000:EF6F	ROM-BIOS
0DH	硬盘/并行口 2 中断	0BA9:03B2	F000:EF6F	ROM-BIOS *
0EH	软盘中断	0BA9:043A	0BF7:043A	DOS STACKS
0FH	打印机/并行口 1 中断	0070:075C	0070:075C	DOS-BIOS

注:XT 型由 DOS STACKS 接管控制权

表9.4　BIOS 软中断

中断号	名　　称	XT 型向量	AT 型向量	控　制　权
10H	视频显示 I/O	F000:FF65	C000:0335	ROM-BIOS *
11H	设备配置检测	F000:F84D	F000:F84D	ROM-BIOS
12H	内存容量检测	F000:F841	C000:1799	ROM-BIOS *
13H	磁盘 I/O	0070:0FC9	0070:1DE3	DOS-BIOS
14H	串行通信 I/O	F000:E739	F000:F739	ROM-BIOS
15H	盒带/多功能实用	F000:F859	F000:F859	ROM-BIOS
16H	键盘 I/O	F000:E82E	F000:E82E	ROM-BIOS
17H	打印机 I/O	F000:EFD2	F000:EFD2	ROM-BIOS
18H	ROM-BASIC	F600:0000	F000:E2C6	ROM-BIOS
19H	磁盘自举	0070:1952	0070:1952	DOS-BIOS
1AH	日时钟/实时钟 I/O	F000:FE6E	F000:FE6E	ROM-BIOS

9.3.2 中断系统的基本职能和结构

- 中断系统的基本功能

(1) 及时记录各种中断请求信号；

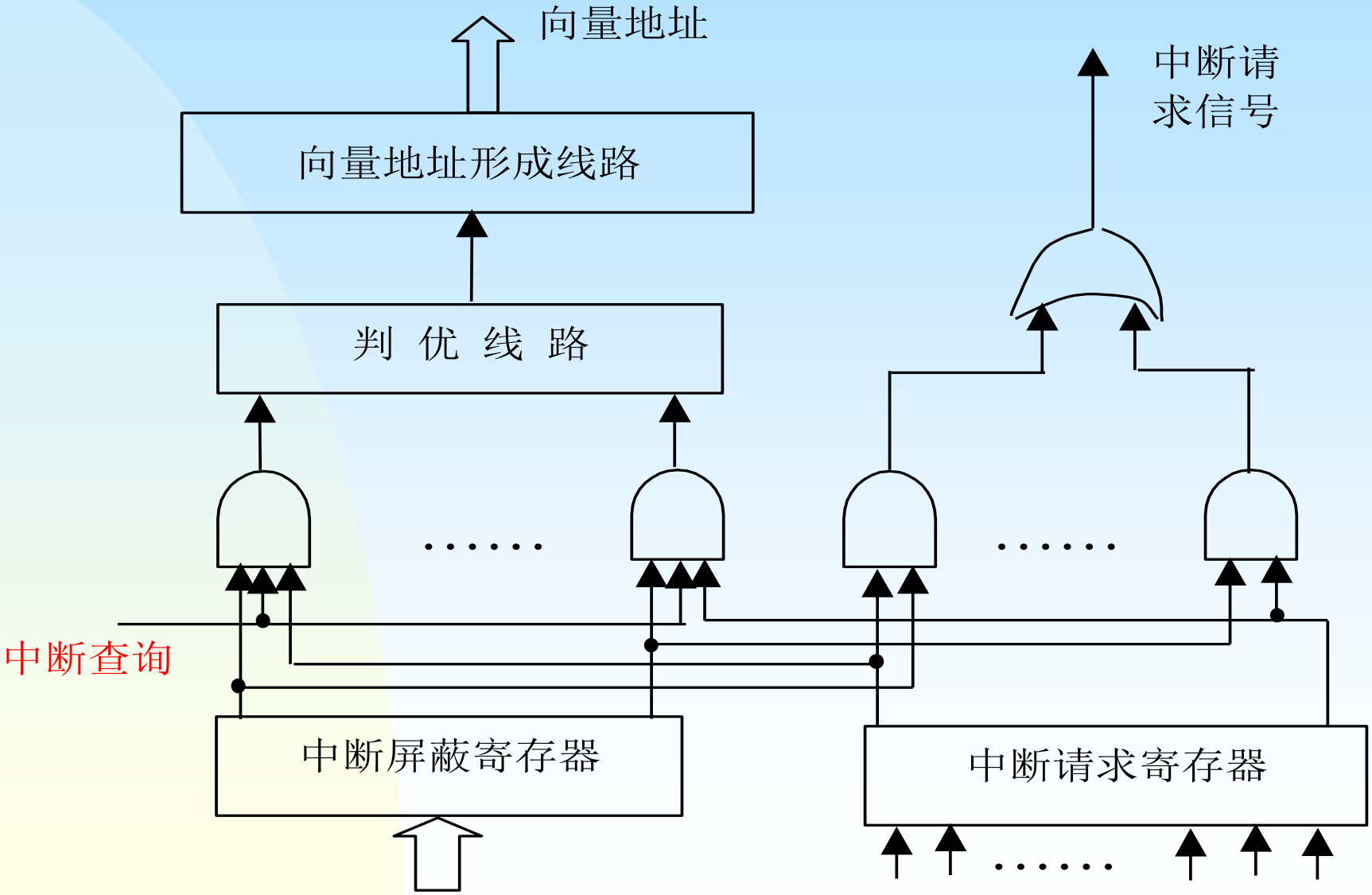
(2) 自动响应中断请求；

(3) 自动判优；

(4) 保护被中断程序的断点和现场；

(5) 中断屏蔽；现代计算机大多采用中断嵌套技术。

中断系统的基本结构



中断嵌套

- 当有新的优先级更高的中断请求发生，CPU立即中止正在执行的中断服务程序，转去处理新的中断。

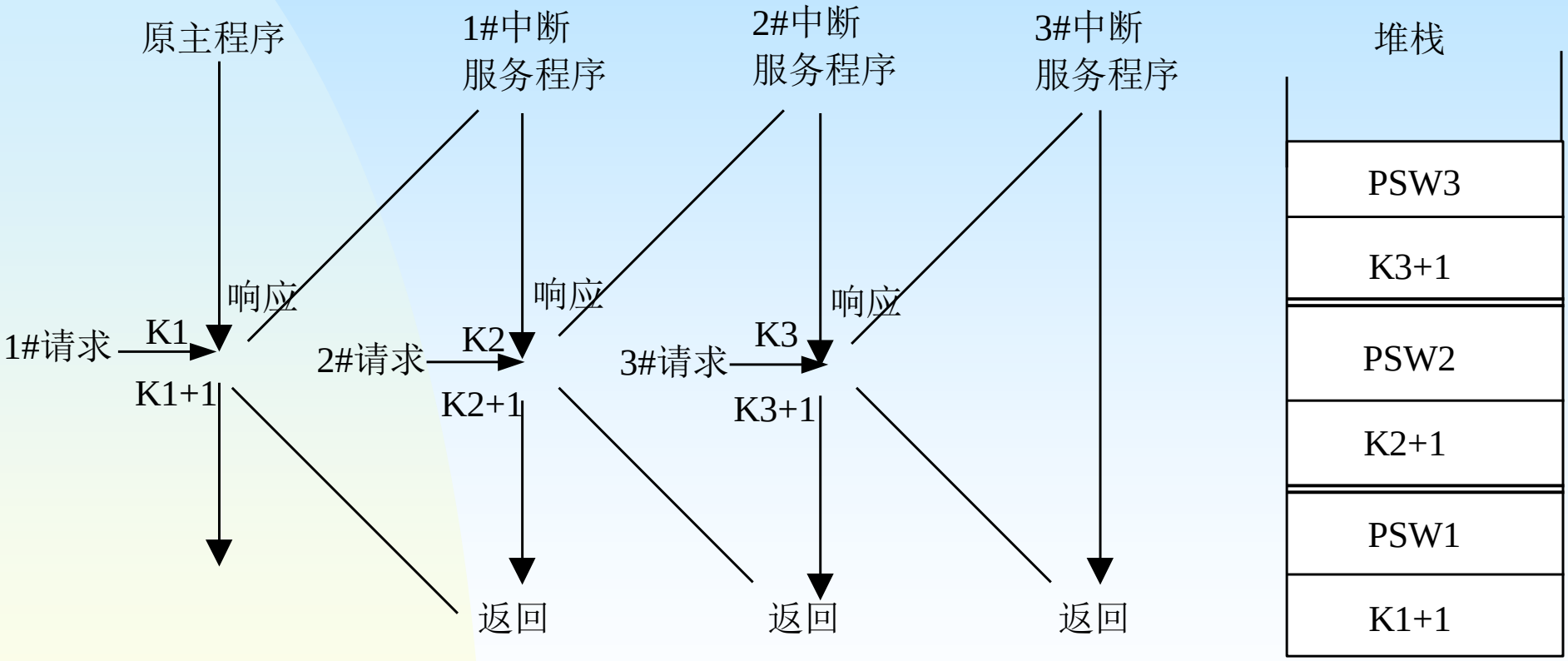
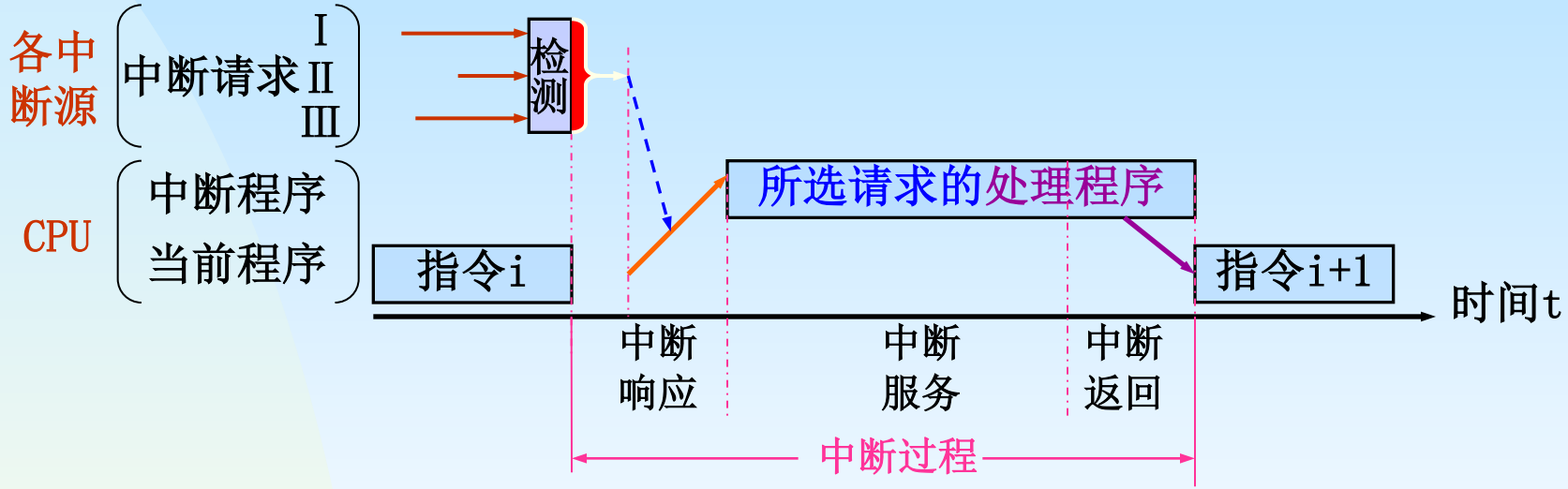


图9.10 中断嵌套过程

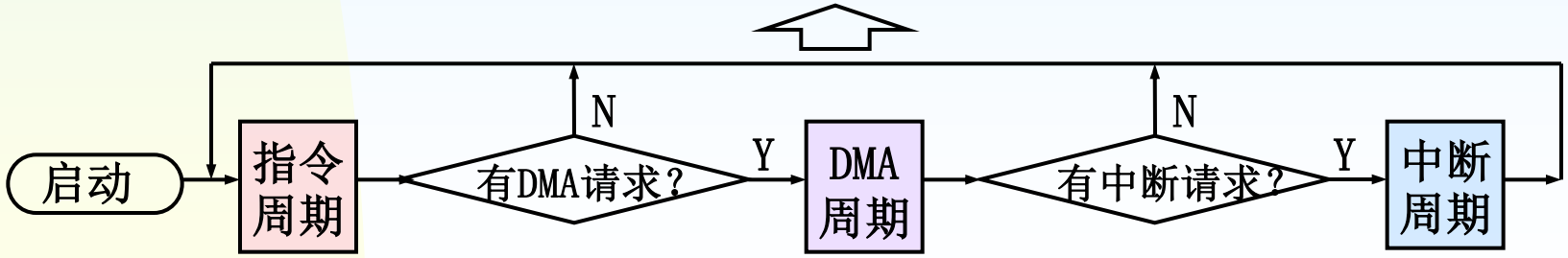
9.3.3 中断过程

中断过程=中断响应+中断服务+中断返回

=中断响应 + 中断处理



CPU工作周期：



- 中断响应阶段和中断处理阶段

1. 中断响应

- 指主机发现中断请求，中止现行程序的执行，到调出中断服务程序这一过程。

（1）保存好程序的关键性信息

- 断点信息保护，硬件自动压栈。

(2) 正确识别中断源

(3) 提高中断响应的速度

- 反映了整个计算机系统的灵敏度。

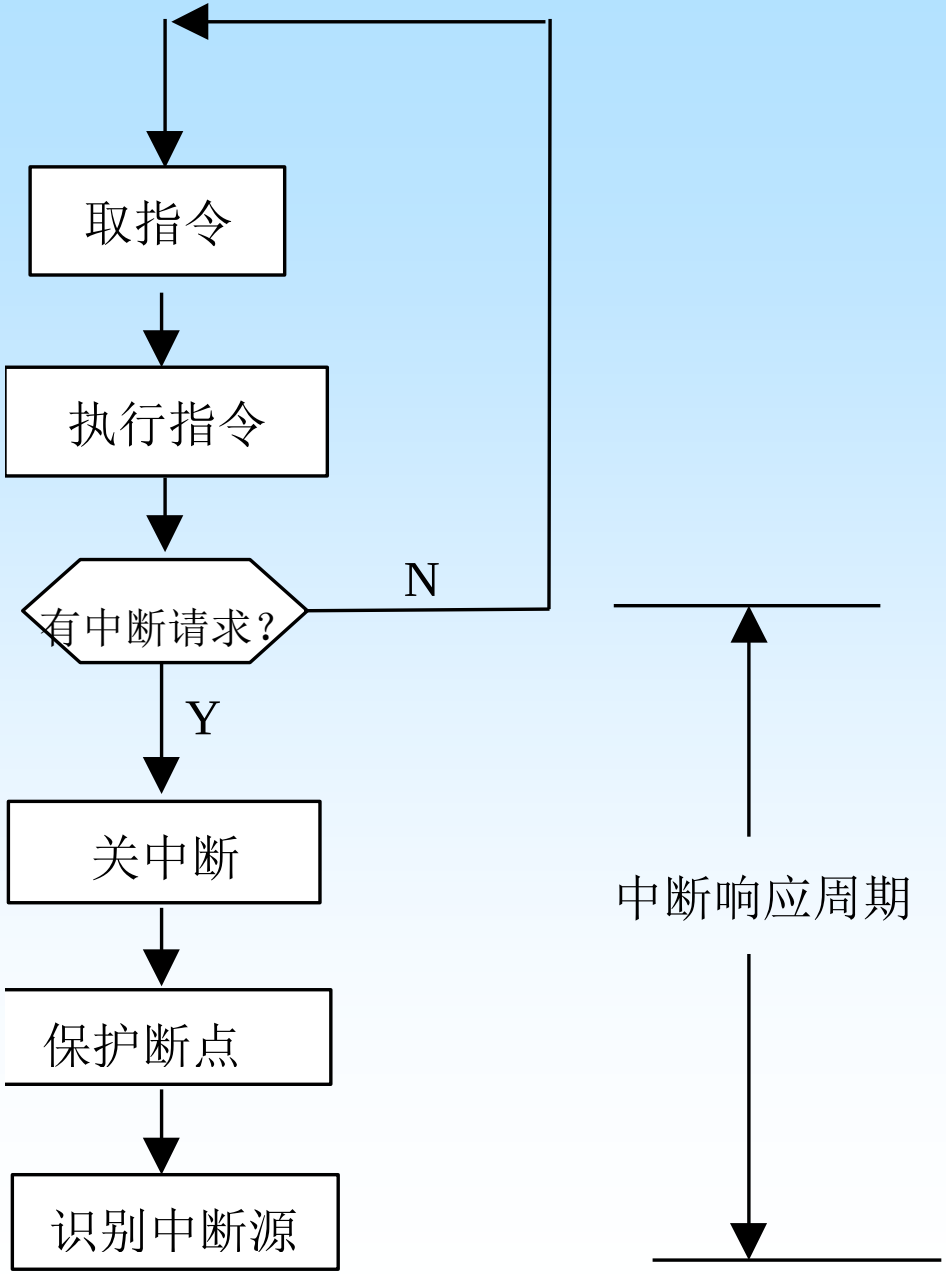
▲ 中断响应的条件

① CPU处于开中断状态（IF=1）；

② 至少要有一个未被屏蔽的中断请求；

③ 在一条指令执行完。

中断响应过程



▲ 中断源的识别方法

① 软件轮询方法

- 查询顺序决定中断优先级。

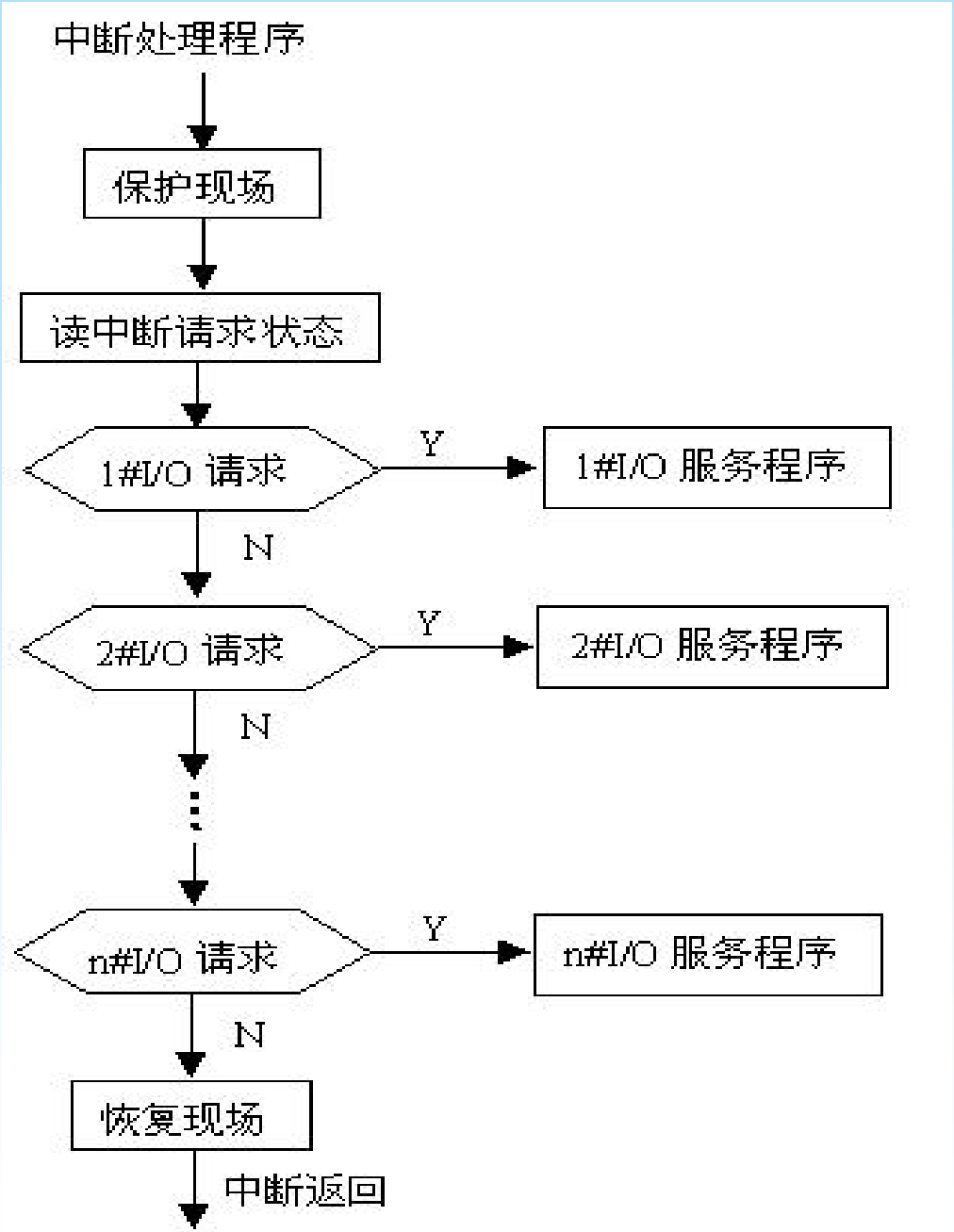


图9.11 中断查询程序的结构

- 硬件结构简单，中断响应慢；CPU的利用率低。

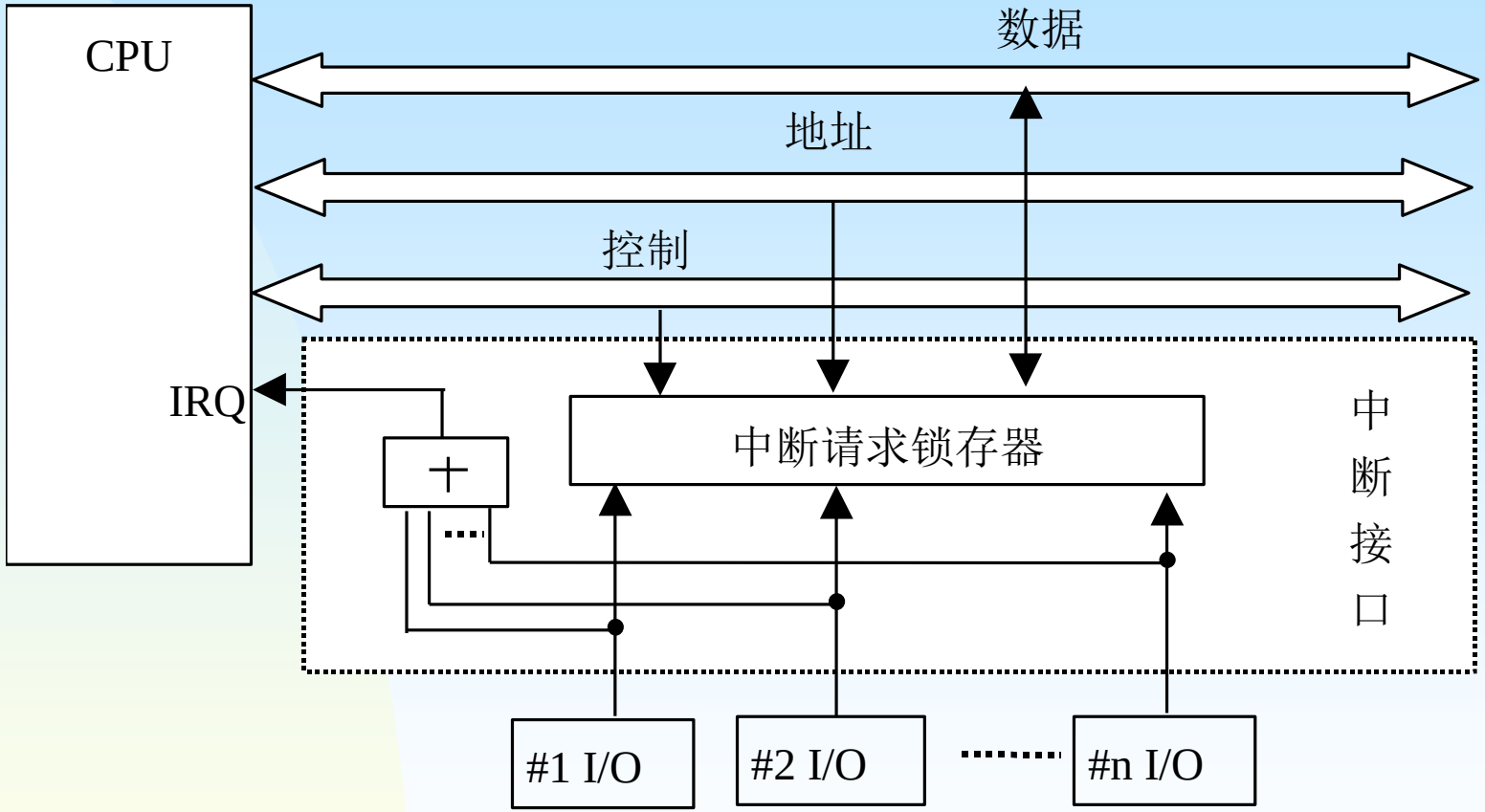


图9.12　程序查询中断的结构

② 硬件判优方法（向量中断）

- 把中断服务程序的首址PC和初始PSW称为中断向量，所有中断向量存放在一个中断向量表中；
- 指向中断向量的指针(地址)称为向量地址(Vector Address)。
- 把中断向量表中与相应中断对应的表项号称为中断类型号。

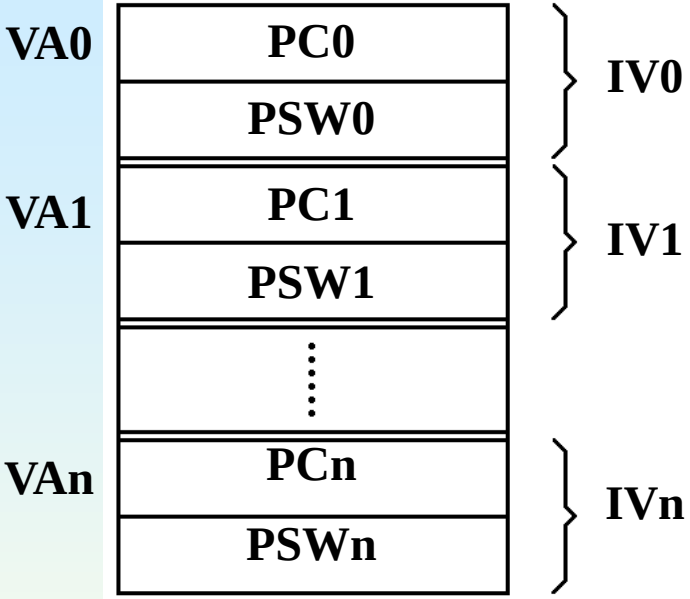


图9.13　中断向量表

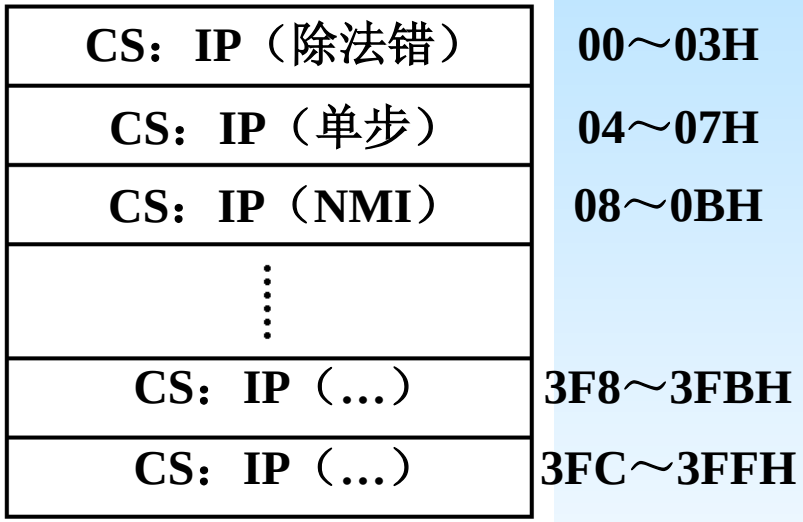
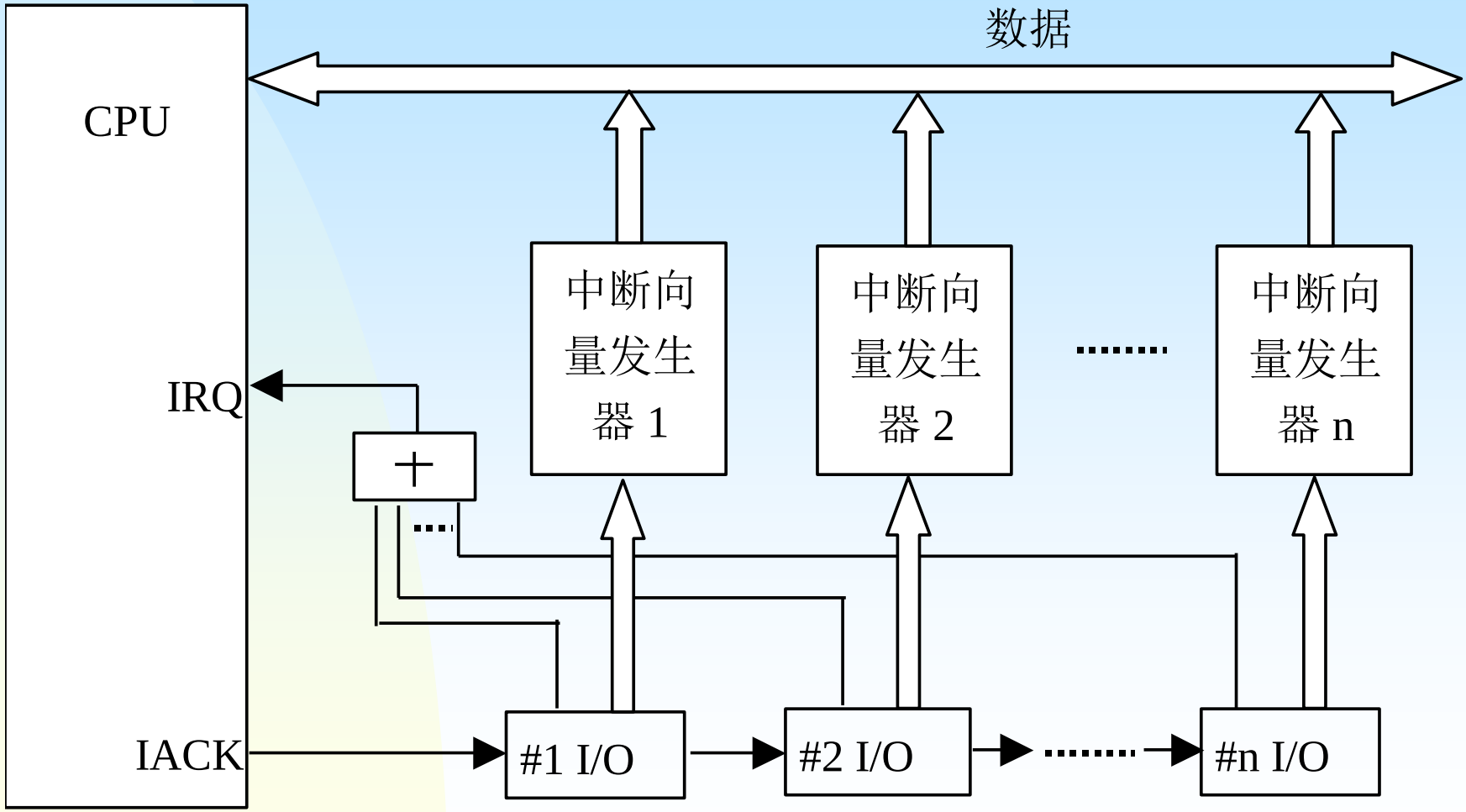


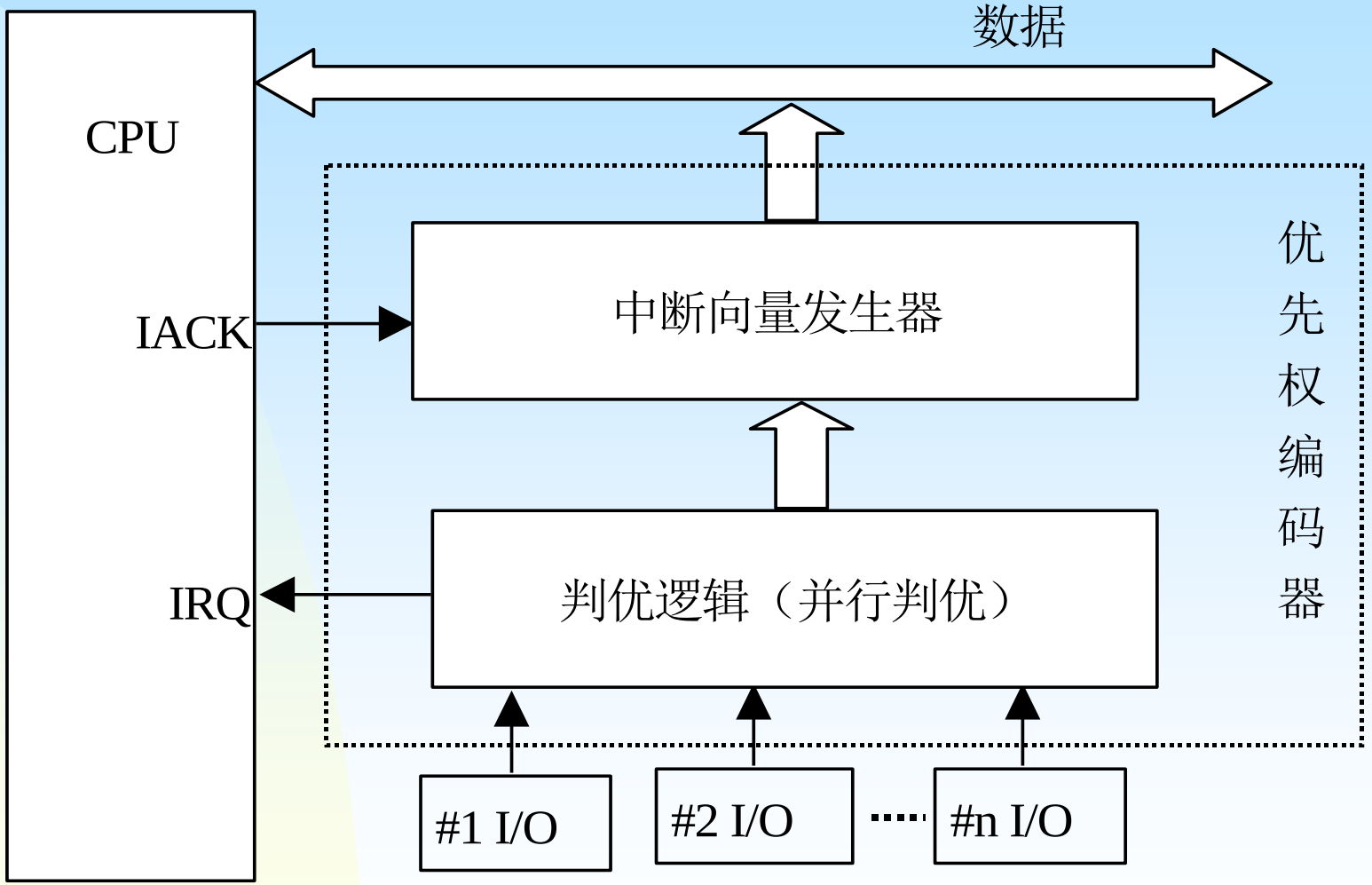
图9.14　8086/8088中断向量表

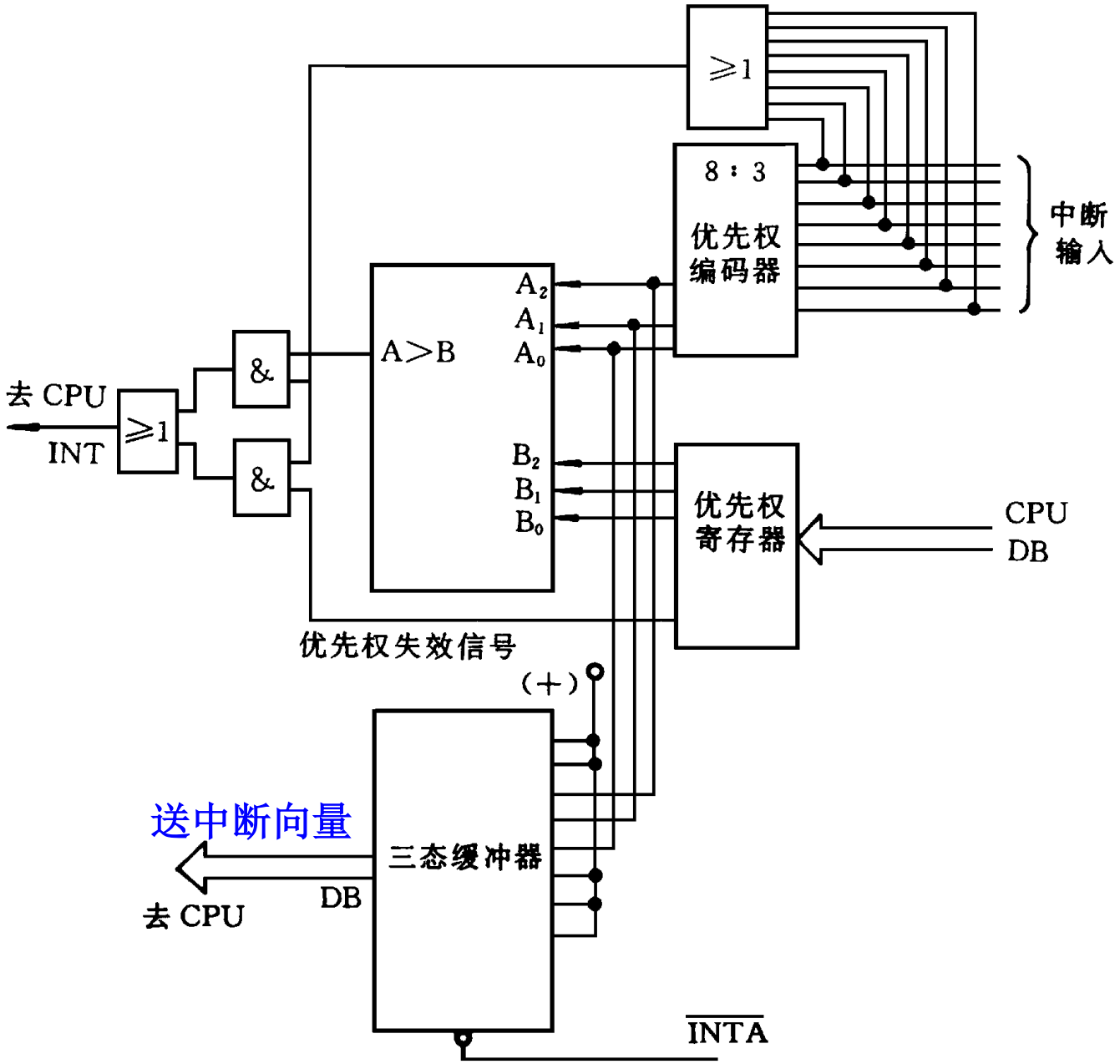
▲ 硬件判优法

① 链式中断查询结构



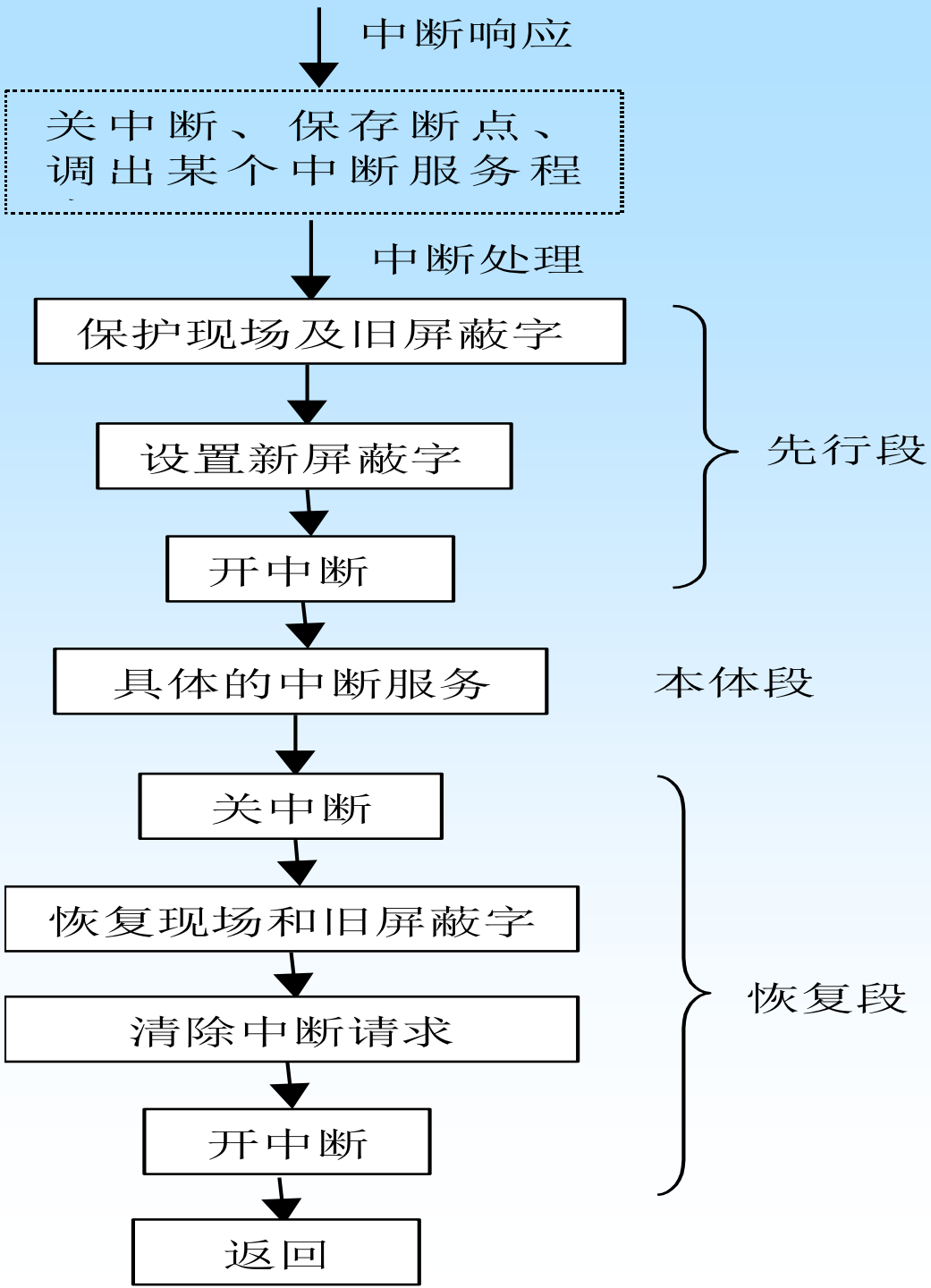
② 多线独立请求中断的结构




≥1
8：3
优先权
编码器
中断
输入
A>B
A2
A1
A0
&
&
去 CPU
≥1
INT
B2
B1
B0
优先权
寄存器
CPU
DB
优先权失效信号
(+)
送中断向量
三态缓冲器
DB
去 CPU
INTA

2. 中断处理

- 执行相应的中断服务程序的过程



▲ 用中断方式控制硬盘和主存储器之间的数据交换

- 例2：假定处理器按500MHz的速度执行，硬盘以 4 字块进行传输，速率为4MB/Sec，假定没有任何数据传输被错过。使用中断驱动I/O，每次传送的开销（包括用于中断响应和处理的时间）是500个时钟周期。如硬盘仅用占处理器5%的时间进行传送，处理器用在数据传送上所花的时间百分比为多少？

- 硬盘每次中断以 4 字块（=16字节）进行传送，传送的速率应达到每秒4MB/16B=250K次中断；
- 每秒钟用于中断的周期数为250K×500=125×10^6，在一次传输中所消耗的处理器时间的百分比为：125×10^6/(500 × 10^6)=25%
- 硬盘仅用其中5%的时间来传送数据，则处理器消耗的平均占用率为25% × 5%=1.25% 。

9.3.4 Pentium中断机制

1. 中断类型

- **Pentium中断源：**

- 外部中断（硬中断）：通过CPU的中断请求线INTR和NMI来实现请求的中断。

  （1）可屏蔽中断：由外设中断源引起的中断通过INTR线进行请求，由IF的值决定是否禁止；

  （2）非屏蔽中断：若发生重要或紧急的硬件故障，如电源掉电、存储器线路错等；不能被禁止。

- 内部异常：通常称为异常中断或称例外，它是由指令执行引发的。

  （1）执行异常：CPU执行一条指令过程中出现错误、故障等不正常条件引发的中断；如被0除，产生溢出中断（类型号为0）；

  （2）执行软件中断指令：如执行INT n指令，将会产生异常中断。

- Pentium共有256种中断和异常；中断类型号（0—255）；

- 中断优先级分为5级；异常中断的优先级高于外部中断的优先级。

2. 中断服务程序进入过程

- 获取中断类型号的途径：

（1）指令给出，如执行软件中断指令INT n，n为中断类型号；

（2）外部提供，可屏蔽中断是在CPU接收到INTR信号时产生一个中断识别周期；

- 非屏蔽中断是在CPU接收到NMI信号时产生，中断类型号固定为2；

（3）CPU识别错误、故障现象，根据异常和中断产生的条件自动指定类型号。

实模式下使用中断向量表

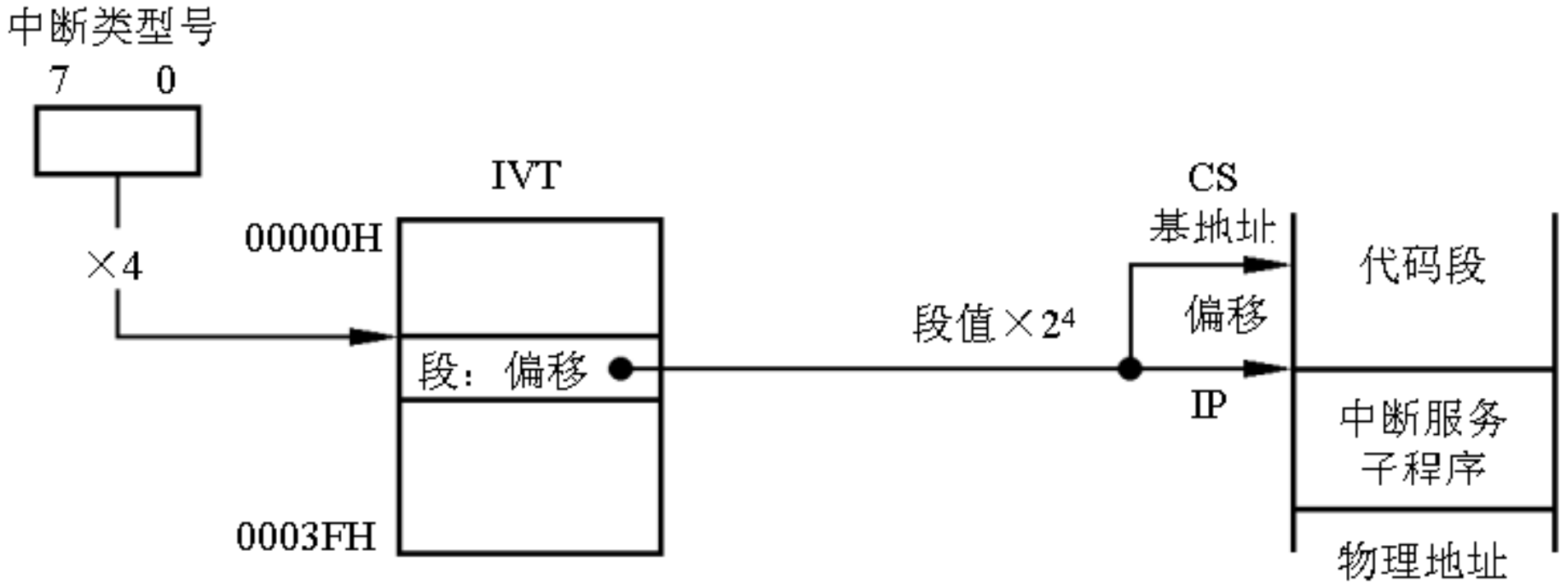


图9.18　实模式下使用中断向量表

保护模式下使用中断描述符表

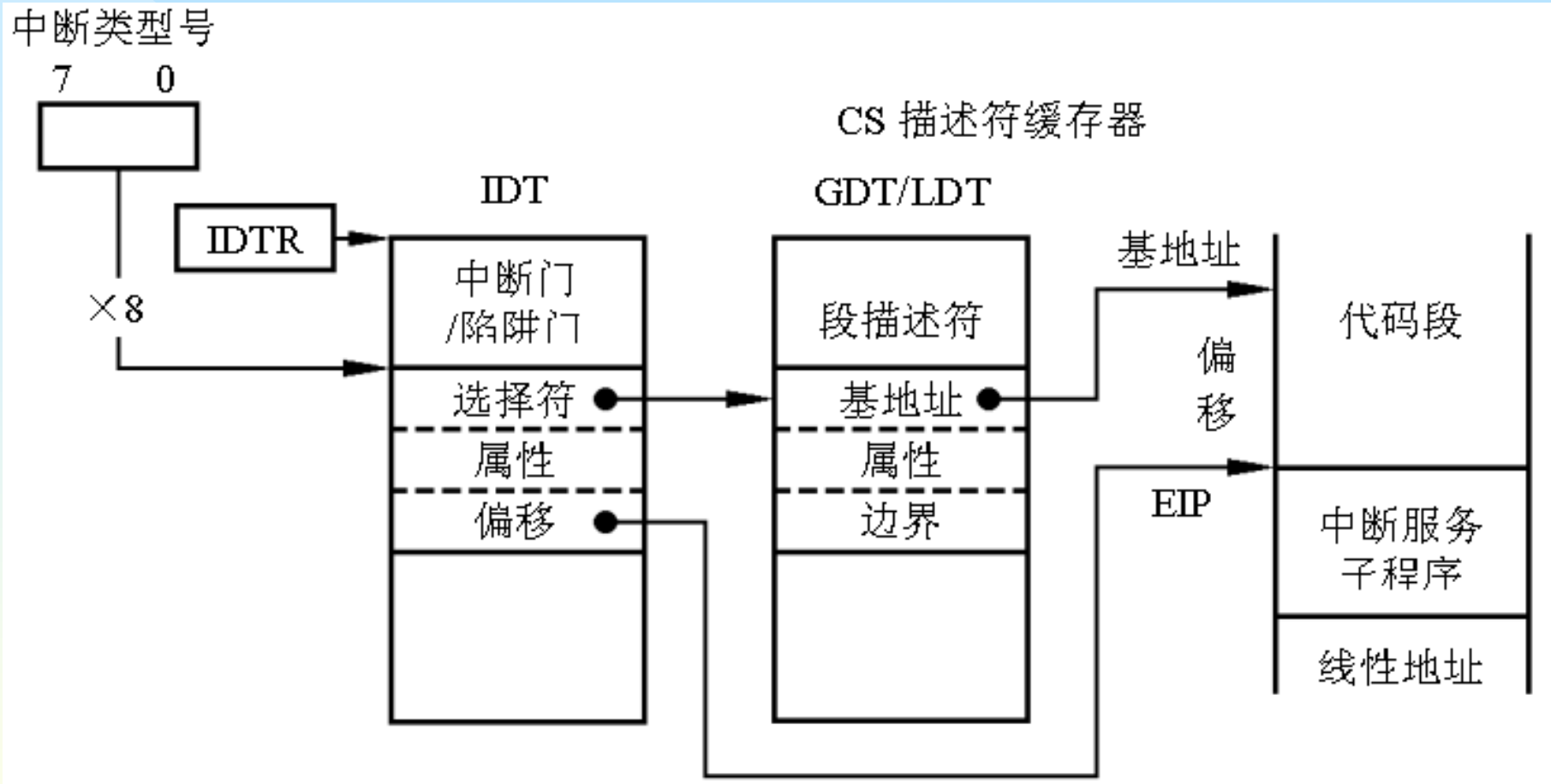


图9.19　保护模式下使用中断描述符表

3. 中断处理过程

1）当进行中断处理的CPU控制权转移涉及到特权级改变时，必须把当前的SS和ESP两个寄存器的内容压入系统堆栈予以保存；

2）标志寄存器EFLAGS的内容也需压入堆栈；

3）清除标志触发器TF和IF；

4）当前的代码段寄存器CS和指令指针EIP也压入此堆栈。

5）如果中断发生伴随有错误码，则错误码也压入堆栈；

6）完成上述中断现场保护后，从中断类型号获取的中断服务程序入口地址（段地址，偏移量）分别装入CS和EIP，开始执行中断服务程序；

7）中断报务程序最后的IRET指令使中断返回；保存在堆栈中的中断现场信息被恢复，并由中断点继续执行原程序。

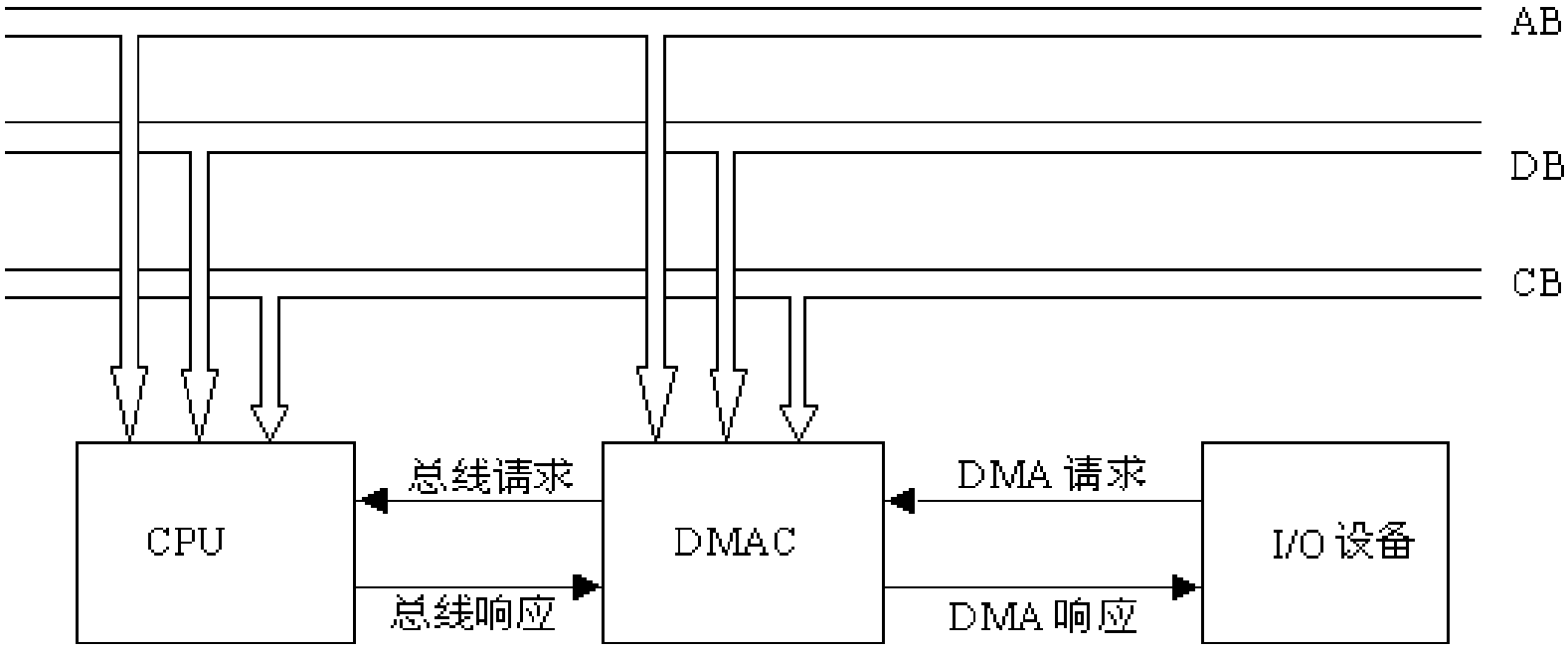
9.4 直接存储器存取(DMA)方式

- **DMA的引入**
  - 程序直接控制方式受“踏步”现象的限制，效率低下，不适合高速设备和主机间的数据传送。
  - 中断控制方式在高速设备、成批数据传送中显出不足

(1) 对I/O请求响应慢；

(2) 数据传送速度慢。

- **DMA方式：**
  - 用专门的**DMA接口硬件**来控制外设与主存间的直接数据交换，**而不通过CPU**。
  - 控制总线进行DMA传送的硬件接口为**DMA控制器**。



DMA传送示意图

- **DMA方式必须以程序查询方式和中断方式为基础。**

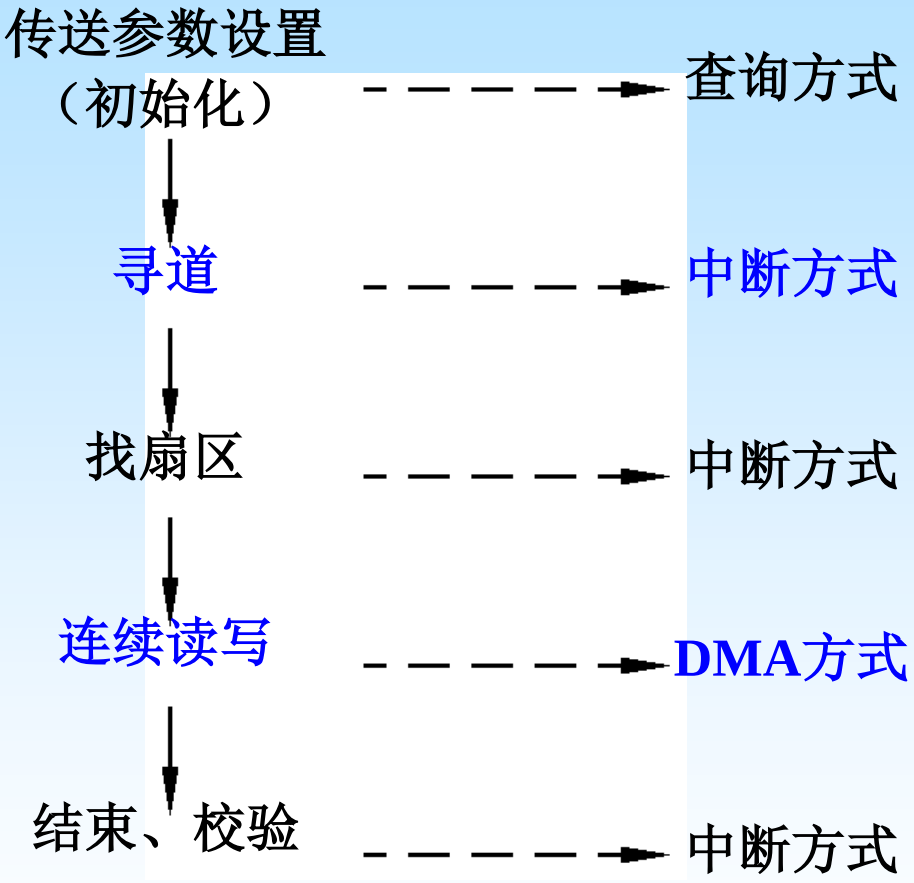


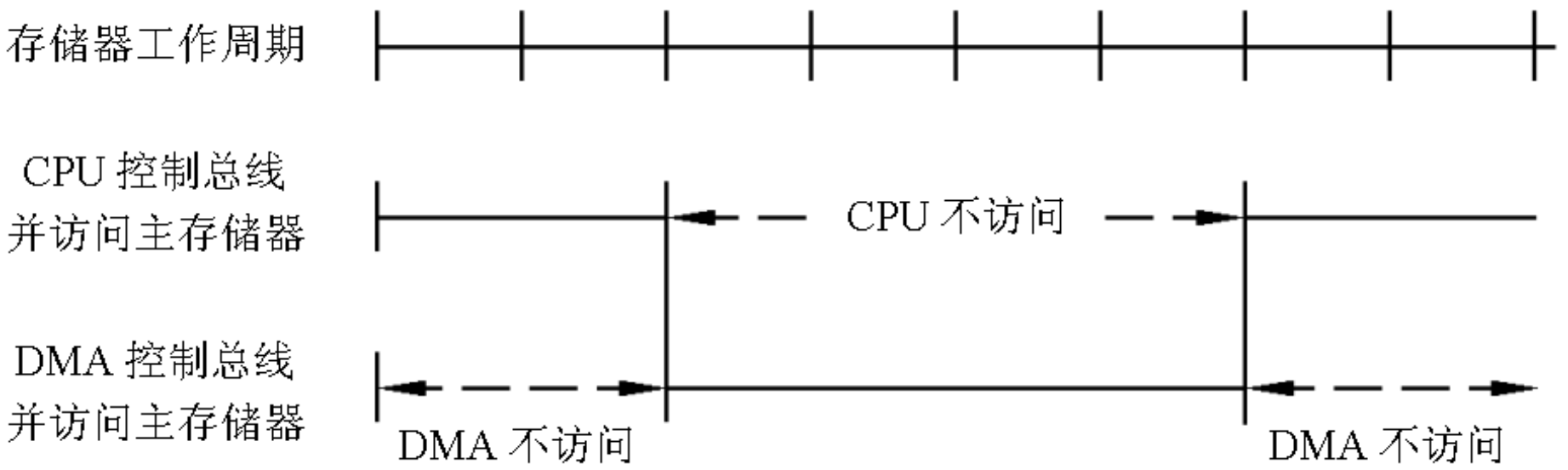
图9.20　采用DMA方式进行磁盘参数传送

- DMA传送方式通常用来高速传送大批量的数据块。如：
  - 硬盘和软盘I/O；
  - 快速通信通道I/O；
  - 多处理机和多程序数据块传送；
  - 在图像处理中，对CRT屏幕送数据；
  - 快速数据采集；
  - DRAM的刷新操作。

9.4.1 DMA工作方式

1. CPU停止法 (成组传送)

- **DMA传输时，CPU脱离总线，停止访问主存，直到DMA传送一块数据结束（该方法CPU工作受影响）。**

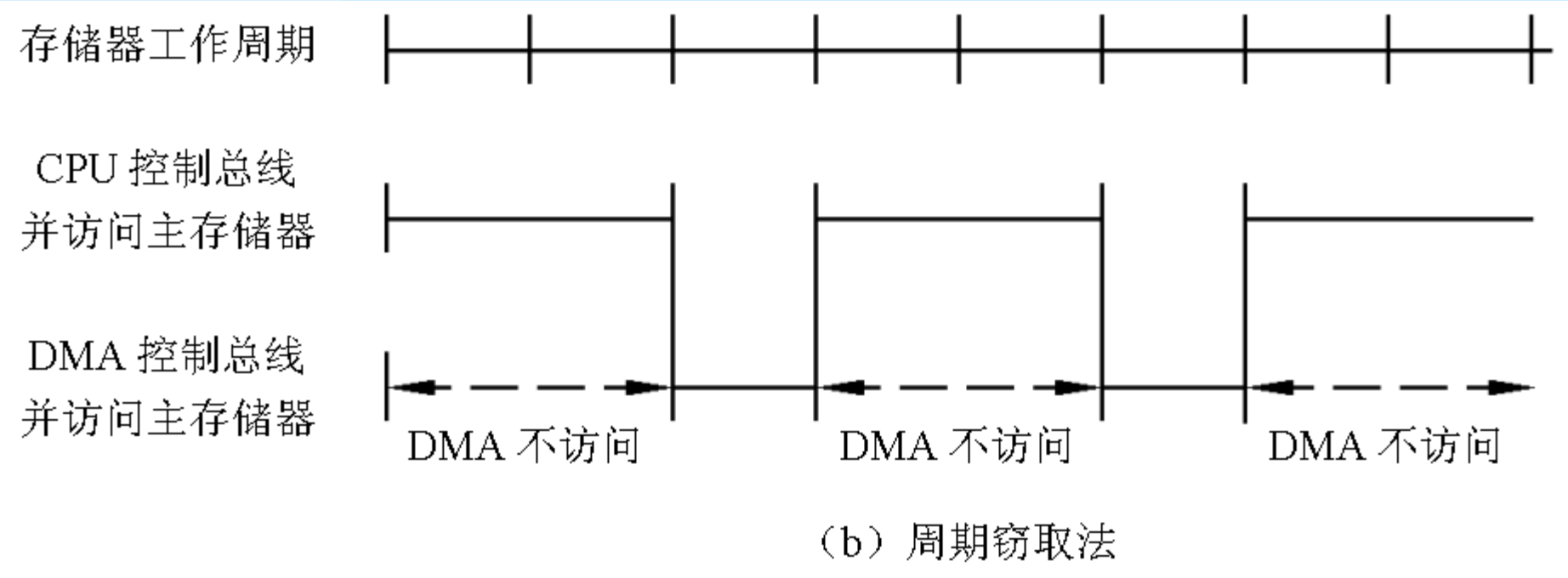


（a）CPU 停止法

- 弥补措施：减少DMA占用总线的时间；I/O设备准备下一数据时，CPU插空访存。

2. 周期挪用(窃取)法 (单字传送)

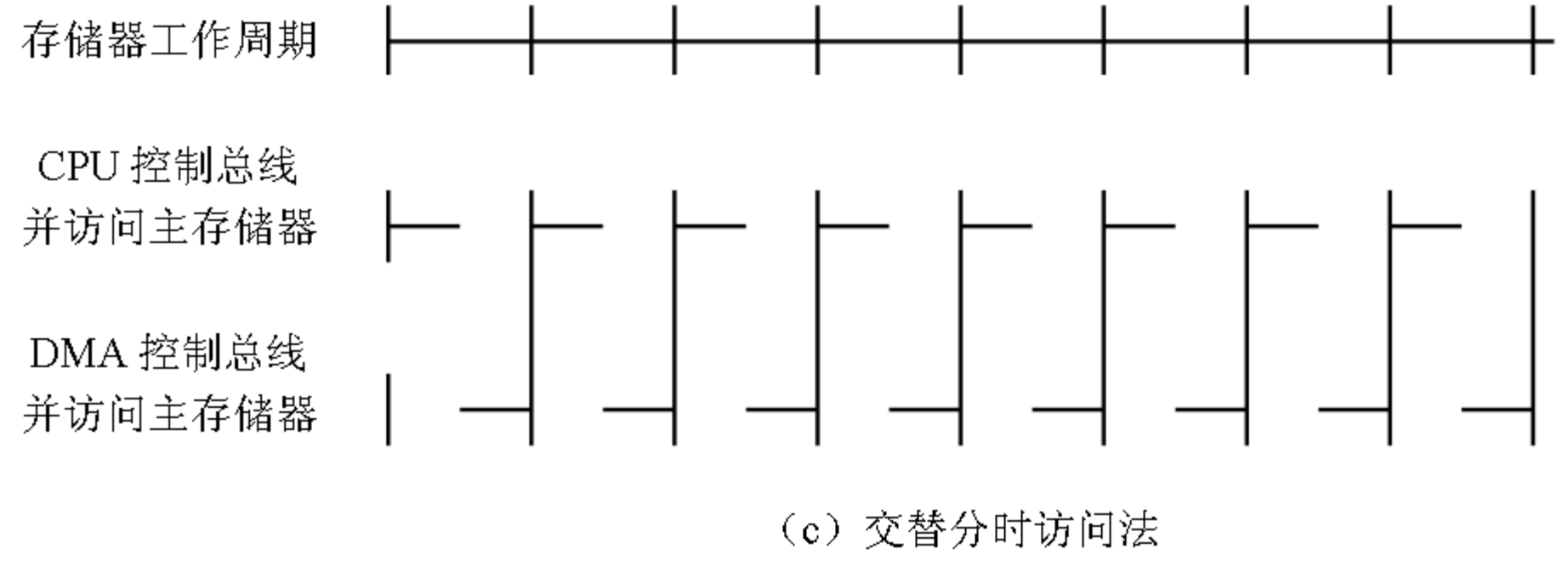
- **DMA传输时，CPU让出一个总线事务周期，由DMA控制器挪用一个主存周期来访问主存，传数据后立即释放总线。**



（b）周期窃取法

3. 交替分时访问法

- 每个存储周期分成两个时间片，一个给CPU，一个给DMA，在每个存储周期内，CPU和DMA都可访问存储器。



（c）交替分时访问法

- DMA不需要总线使用权的申请和释放。

- **I/O设备要求进行DMA传送会遇到三种情况**

1) CPU不需访问主存

- **CPU和DMA不发生冲突，两者并行。**

2) CPU正在访问主存

- **须等到存储周期结束后，CPU让出总线，DMA才能访存。**

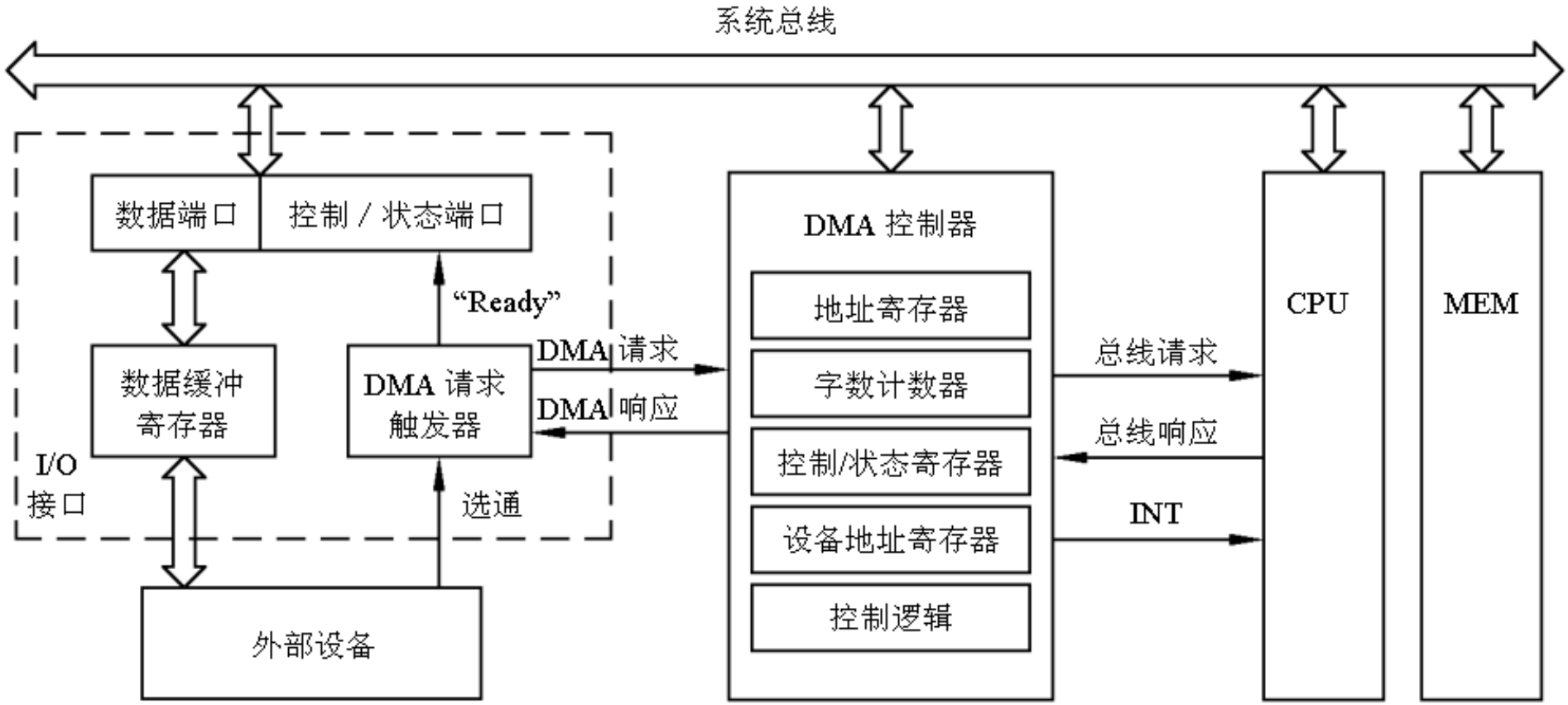
3) CPU也同时要访问主存

- **先让DMA占用总线，窃取一个主存周期，完成数据交换。**

9.4.2 DMA接口的结构和功能

- **DMA控制器（ DMAC）**
  - **CPU把要传送的数据个数、数据块在内存的首址、数据传送的方向、设备的地址等参数送给DMA控制器（ DMAC初始化时，DMAC处于被动状态）；**
  - **启动外设进行数据准备工作，而I/O设备和主存交换数据的事情就交给了DMA控制器（DMAC处于主动状态） 。**

DMA接口的典型结构


系统总线
数据端口
控制 / 状态端口
数据缓冲
寄存器
DMA 请求
触发器
“Ready”
DMA 请求
DMA 响应
I/O
接口
选通
外部设备
DMA 控制器
地址寄存器
字数计数器
控制/状态寄存器
设备地址寄存器
控制逻辑
总线请求
总线响应
INT
CPU
MEM

- **DMA接口功能**

(1) 接收外设的“DMA请求”信号，向CPU发“总线请求”信号。

(2) 当CPU发出“总线响应”信号后，接管对总线的控制。

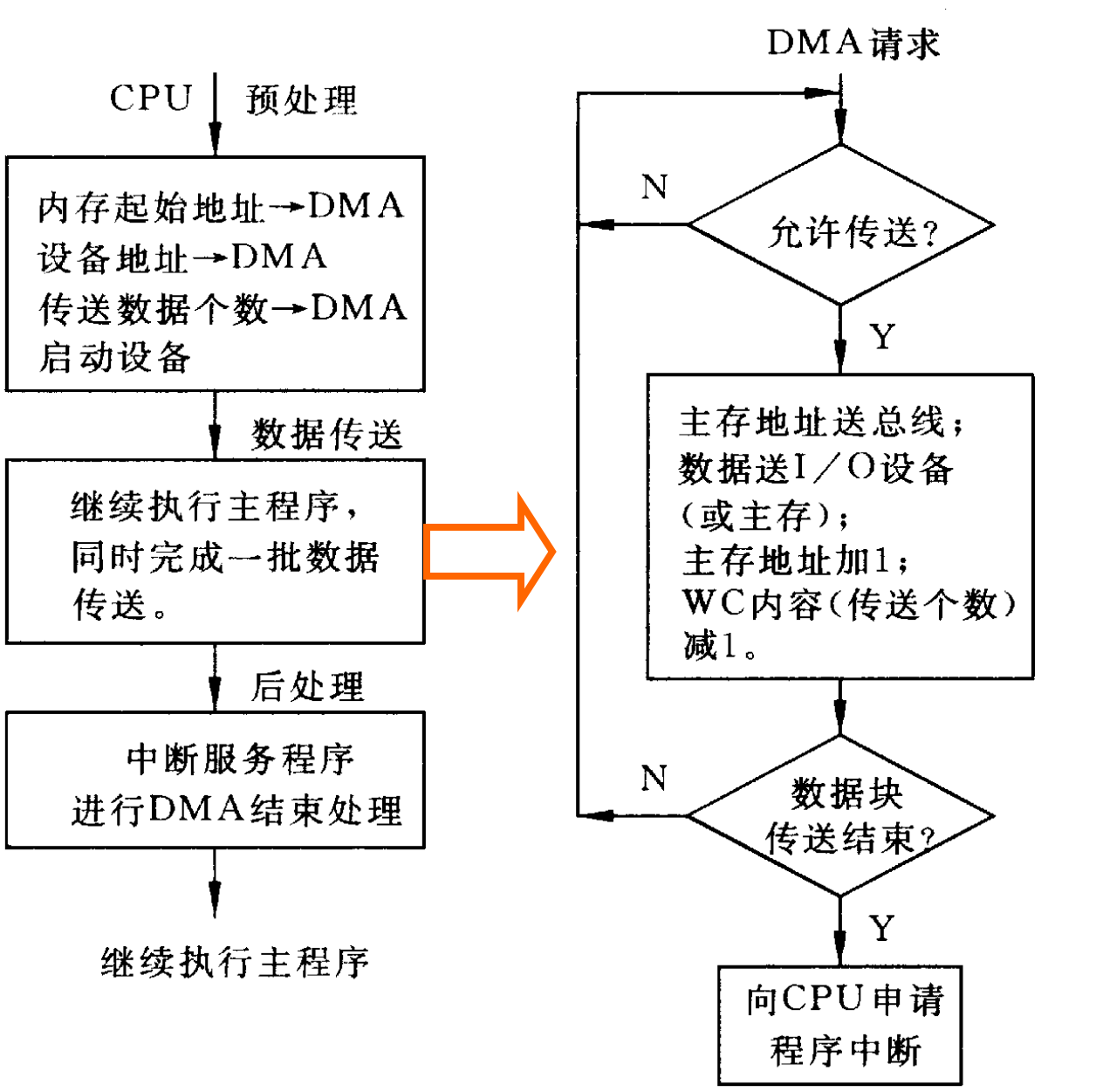
(3) 在地址线上给出主存地址，并自动修改主存地址。

(4) 识别传送方向以在控制线上给出正确的读写控制信息。

(5) 确定传送数据的字节个数。

(6) 发出DMA结束信号。引起一次DMA中断，进行数据校验等一些后处理。

9.4.3 DMA数据传送过程



（a）数据传送的三个阶段　　（b）第二阶段的数据传送过程

图9.21　DMA 数据传送过程

- 例1:假设处理器按500MHz的速度执行。硬盘以 4 字块进行传输，速率为4MB/Sec，假定没有任何数据传输被错过。对于DMA传送的初始化设置，假定处理器花了1000个时钟周期，并且在DMA完成后中断的处理需要500个时钟。如果从硬盘发出的平均传输量为8KB（即每次DMA传送8KB的数据块）。如果硬盘进行传送的时间占100%，那么具有500MHz的处理器在硬盘I/O操作上的花销是多少？

  解：

- 处理器一秒钟内有4MB / 8KB =0.5×10^3个 DMA传送，
- 一秒钟内CPU花在DMA传送上的开销为：

  0.5×10^3×(1000+500)=750×10^3个时钟周期。
- 在硬盘I/O操作上处理器花费的时间占750×10^3/500×10^6 =1.5×10^{-3}=0.15%。
- 中断传送处理器消耗的平均占用率为1.25% 。

- 当DMA方式被用于硬盘接口时
  - 在数据传送期间不消耗处理器周期，即使硬盘一直在进行I/O操作，CPU为它服务的时间仅占0.15%；
  - 事实上，硬盘在大多数时间内并不进行数据传送。因此处理器的占用率会更低；
  - 如果处理器同时要竞争存储器的话，因存储器忙于进行DMA传送，处理器将会被延迟与存储器交换数据；
  - 通过使用Cache，处理器可以避免大多数访存冲突。一般存储器带宽的大部分都可以让给I/O设备使用。

9.4.4 DMA与存储器系统

- DMA引入到I/O系统中时，存储器系统和处理器之间的关系发生改变，这在虚拟存储器系统和Cache 系统中会产生一些问题。
  - 若用虚拟地址，则DMA接口必须要将虚拟地址转换为物理地址；而使用物理地址，则每次DMA传送不能跨页。
- 问题的解决要结合硬件和软件两方面的技术支持
  - DMA接口中安排一个小的类似页表的地址映射表，用于将虚拟地址转换为物理地址；DMA初始化时，由操作系统进行地址映射。
  - 另外一种方法是操作系统把跨页传送分解成多次小数据量传送，每次只限定在一个物理页面内。

◆ I/O数据的一致性问题

- 采用Cache的系统中，一个数据项可能会产生两个副本，一个在Cache中，一个在存储器中。DMA控制器直接向存储器发出访存请求而不通过Cache，这时，DMA看到的一个主存单元的值与处理器看到的Cache中的副本可能不同。

▲ 解决I/O数据的一致性问题的三种方法

（1）让I/O活动通过Cache进行，保证了在I/O读时能读到最新的数据，而I/O写时能更新Cache中的数据（会影响Cache的命中率，对处理器的性能也带来很多负面的影响）。

（2）让操作系统在I/O读时有选择地使某些Cache块无效，而在I/O写时迫使Cache进行一次回写操作，称Cache刷新。

（3）通过一个硬件机制来选择被刷新或使无效的cache项；常被用在多处理器系统中。

作业：P293— 9、13、16、26