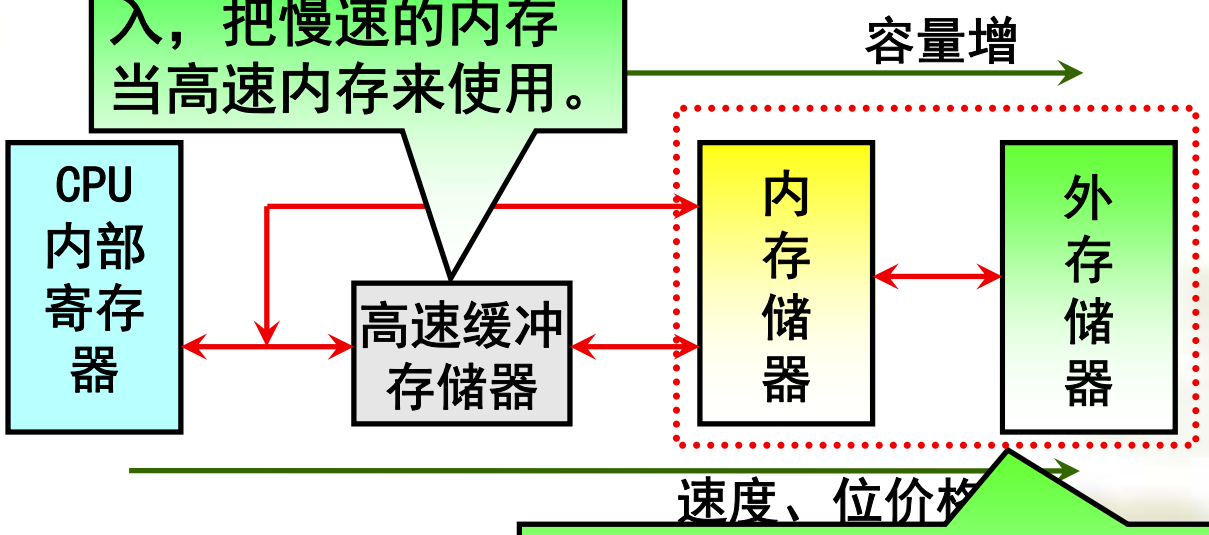
5.1 现代高档微机系统的存储器体系结构

现代高档微机系统中，存储器技术的发展始终是以实现低成本、大容量和高速度为其追求目标，而用单一工艺制造的半导体存储器往往难以同时满足这三方面的要求。为解决这一矛盾、提高存储器系统的性能，目前高档微机系统普遍采用以下结构来组织整个存储器系统：

- 分级存储器结构
- 虚拟存储器结构

1. 分

高速缓存的引入，把慢速的内存当高速内存来使用。

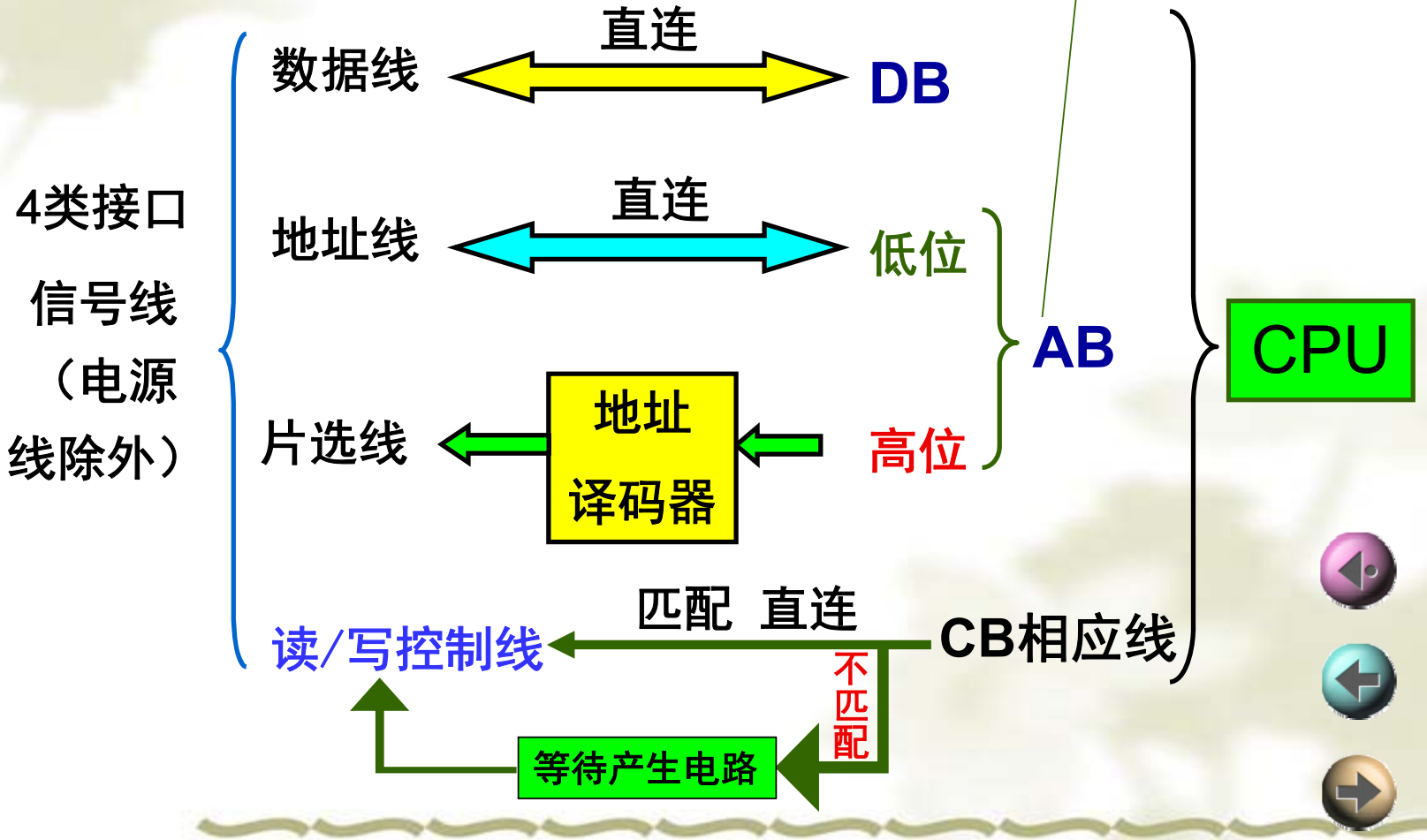


2. 虚拟存储器结构

虚拟存储器技术是在内存与外存之间引入相应的硬件和软件，把大容量的外存当大容量的内存来使用。

2. 与CPU的连接特性

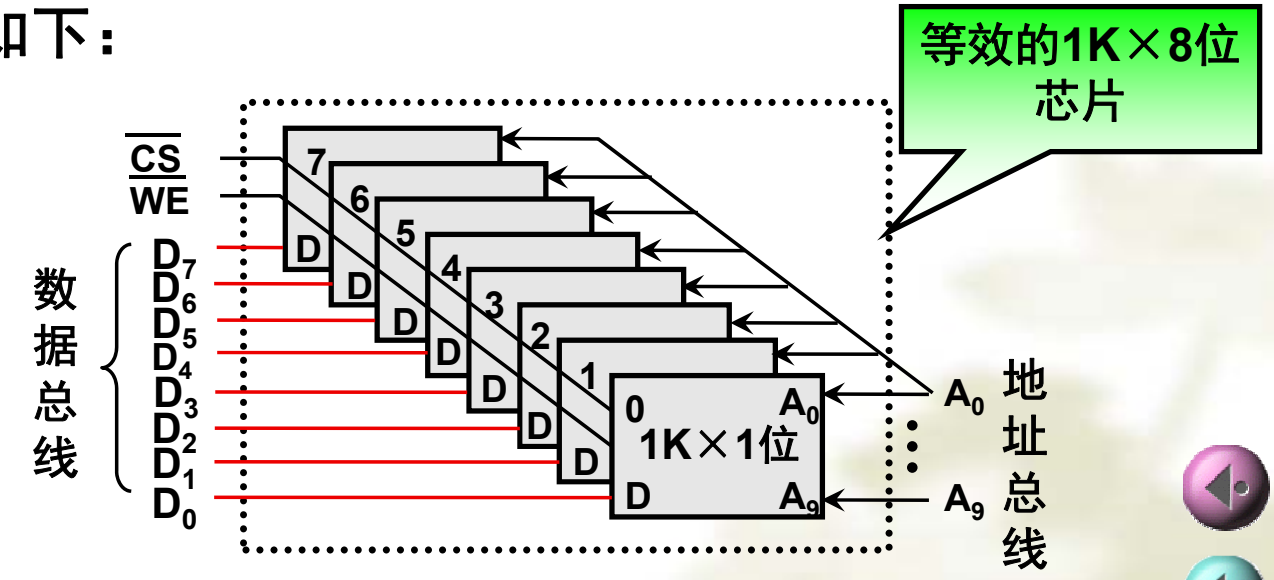
关键：高低位AB如何划分



1）位 扩 展

通过位扩展，满足（8位）字长要求。

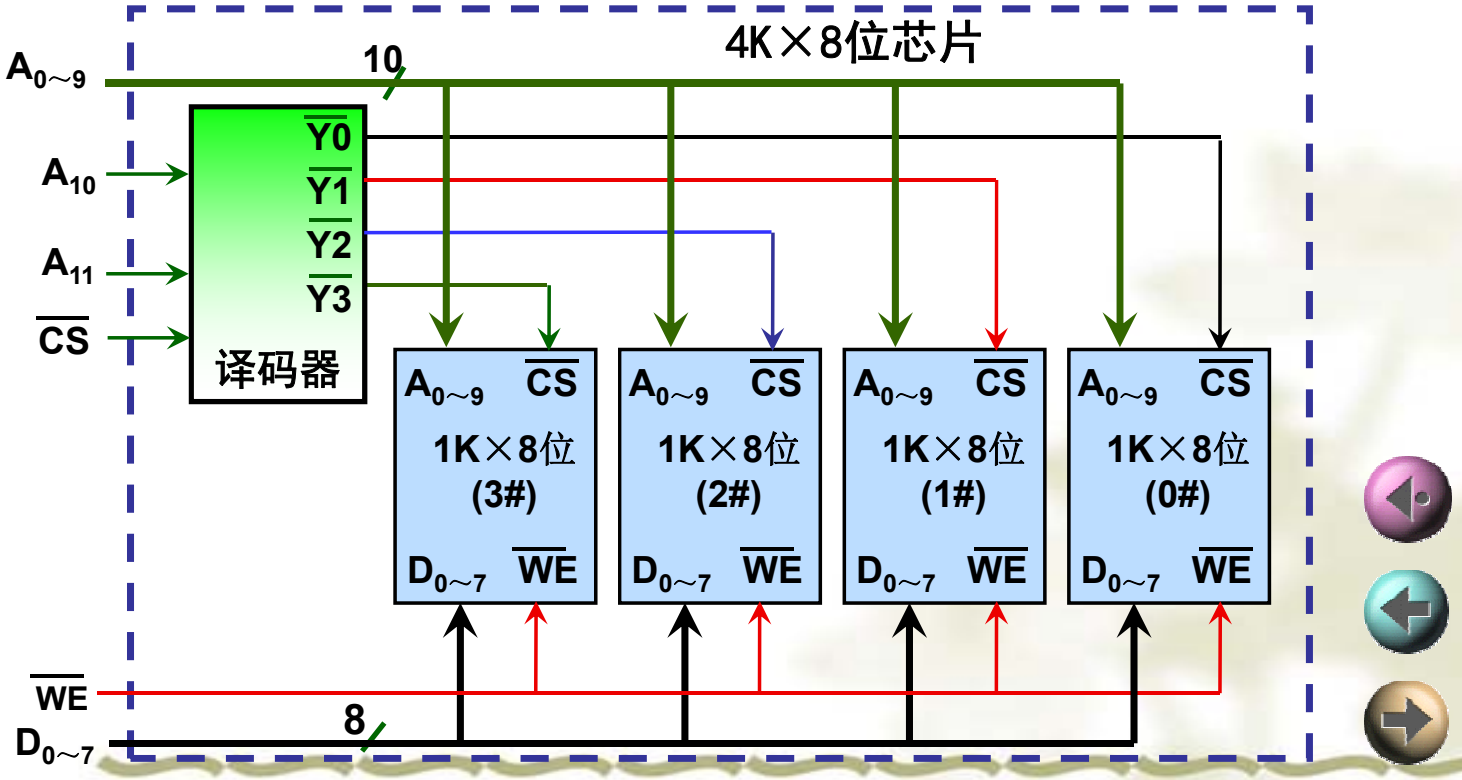
例如，用1K×1位芯片组成1KB存储器的位扩展设计如下：



- 地址、片选、读/写控制线并连
- 数据线分连

2) 字 扩 展 ——通过字扩展，满足字数要求。

- 地址线、数据线、读/写等控制线并连
- 片选线分连

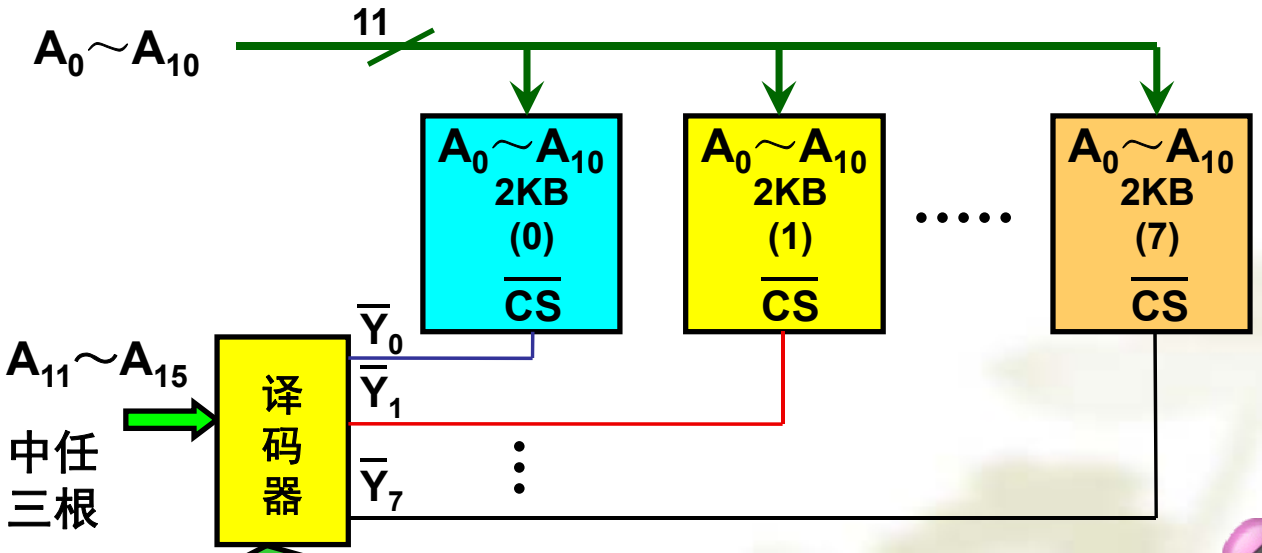


①线选法

②局　部译码法

③全　局译码法

对余下高位地址总线中的一部分进行译码，译码输出作为各存储器芯片的片选控制信号。



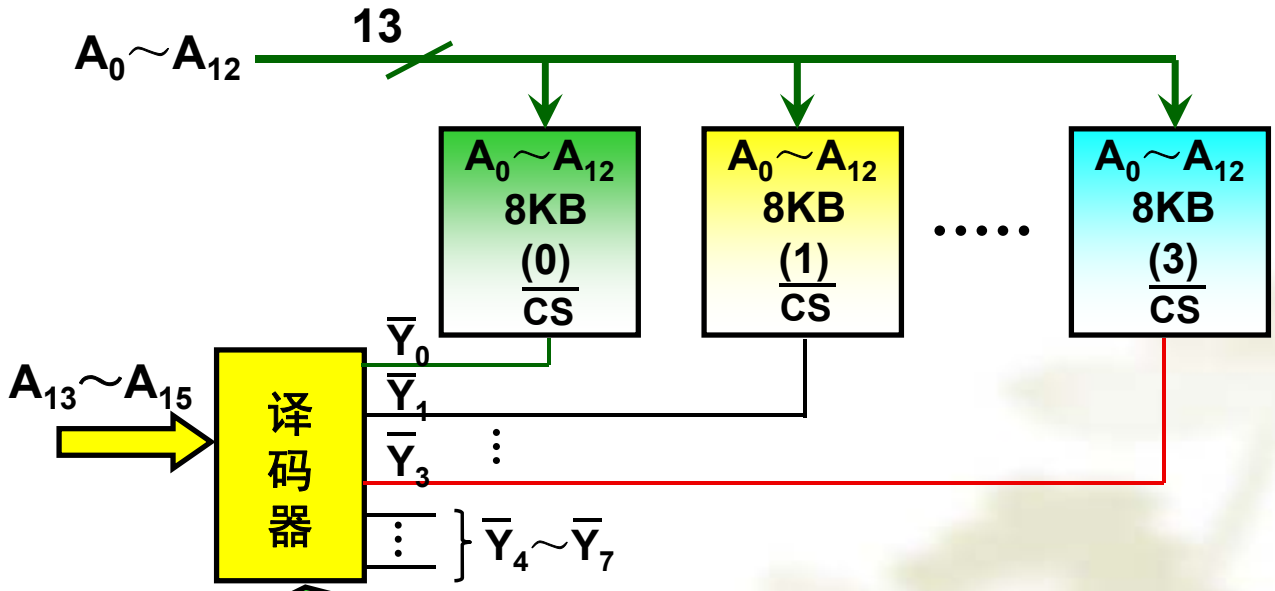
部分高端地址线未参与译码，也存在地址重叠和地址不连续问题，一般在线选法不够用，而又不需要全部地址空间时使用，以简化译码电路。

①线选法

②局　部译码法

③全　局译码法

对余下高位地址总线全部译码，译码输出作为各存储器芯片的片选控制信号。



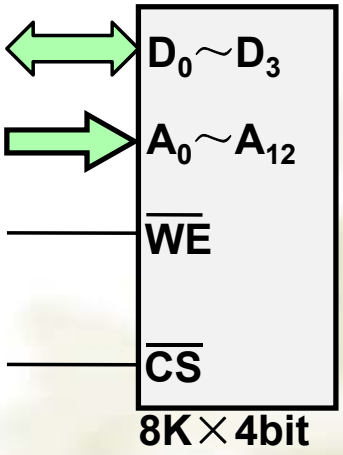
无论是局部译码还是全译码，译码方案既可采用门电路译码、译码器芯片译码，还可采用PROM芯片译码等。

最复杂。

2. 字位全扩展设计实例

例5.2 试用8K×4位的SRAM芯片为某8088微机系统构成一个16KB的RAM存储器，RAM的起始地址为90000H。

解：该例SRAM芯片字长不足8位，需用2个芯片为一组进行位扩展后，再进行字扩展。



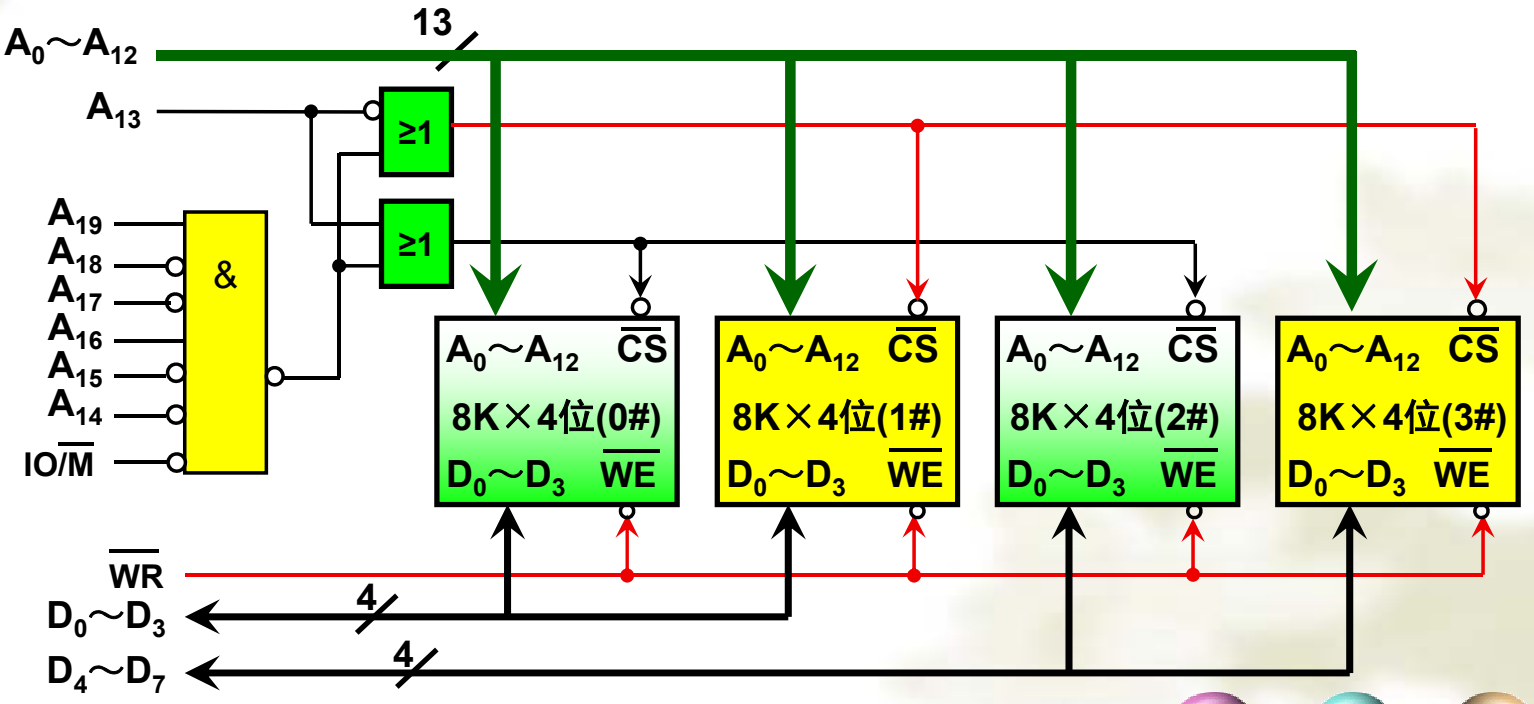
(1) 列出各芯片组的地址范围和存储器地址位分配

芯片组	位分配		地址范围
	$A_{19}A_{18}A_{17}A_{16}A_{15}A_{14}A_{13}$	A_{12} ～ A_0	
0#、2#	1 0 0 1 0 0 0	0000～1FFFH	90000～91FFFH
1#、3#	1 0 0 1 0 0 1	0000～1FFFH	92000～93FFFH

← 选芯片 → ← 选单元 →

(2) 接口电路

假定用门电路译码，则字位扩展设计如下：



5.3.1 半导体存储器的分类

半导体存储器从功能和应用角度主要有两大类：

- ROM的类型
  - 掩模ROM
  - PROM
  - EPROM
  - E^2PROM
  - Flash ROM
- RAM的类型
  - SRAM
  - DRAM
  - IRAM
  - NVRAM

半导体存储器的主要性能指标

1. 存储容量

- 存储单元个数×每个存储单元的位数

2. 存取时间和存取周期

- 存取时间：实现一次读/写所需要的时间
- 存取周期：连续启动两次独立的存储器操作所需间隔的最小时间

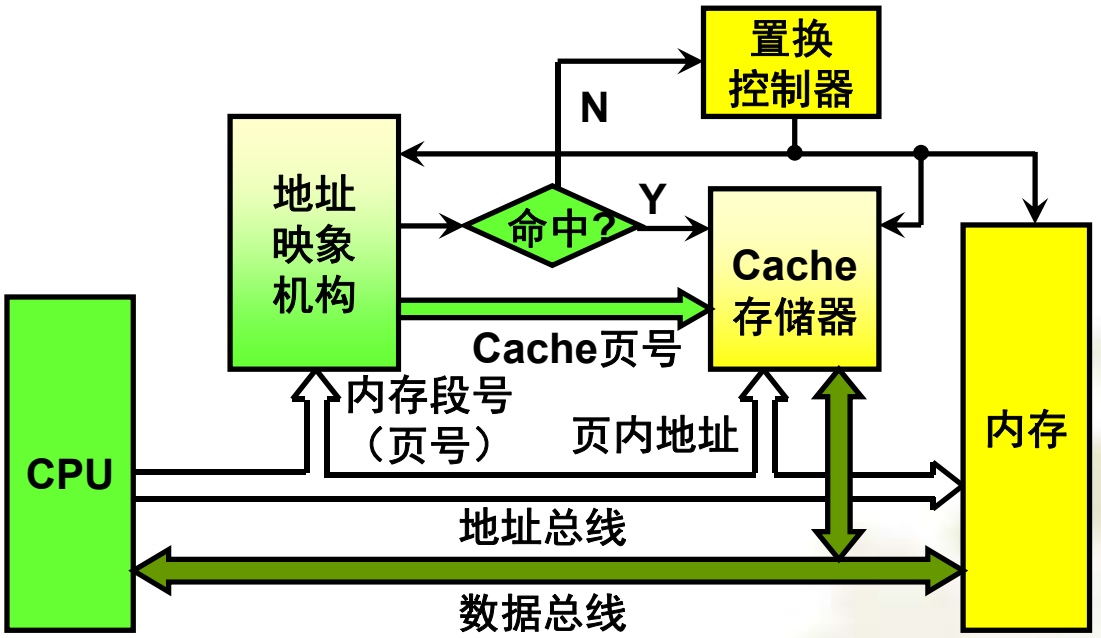
3. 可靠性

- 存储器的可靠性用MTBF平均故障间隔时间来衡量
- MTBF：两次读写错误之间的间隔时间
- 存储器常采用纠错编码技术延长MTBF提高可靠性

4. 功耗

- 目前发展趋势是向低功耗发展，提高系统稳定性

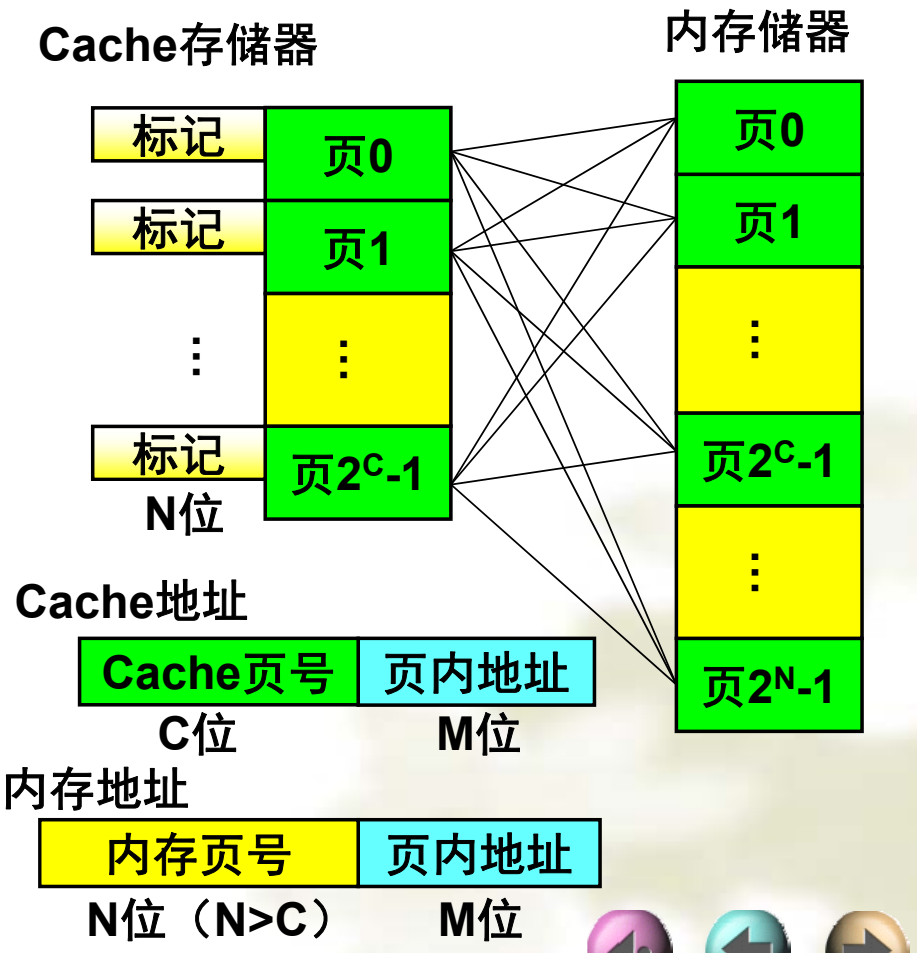
5.4.1 Cache的基本结构和工作原理



动画演示　　Cache存储器结构

1. 全关联方式

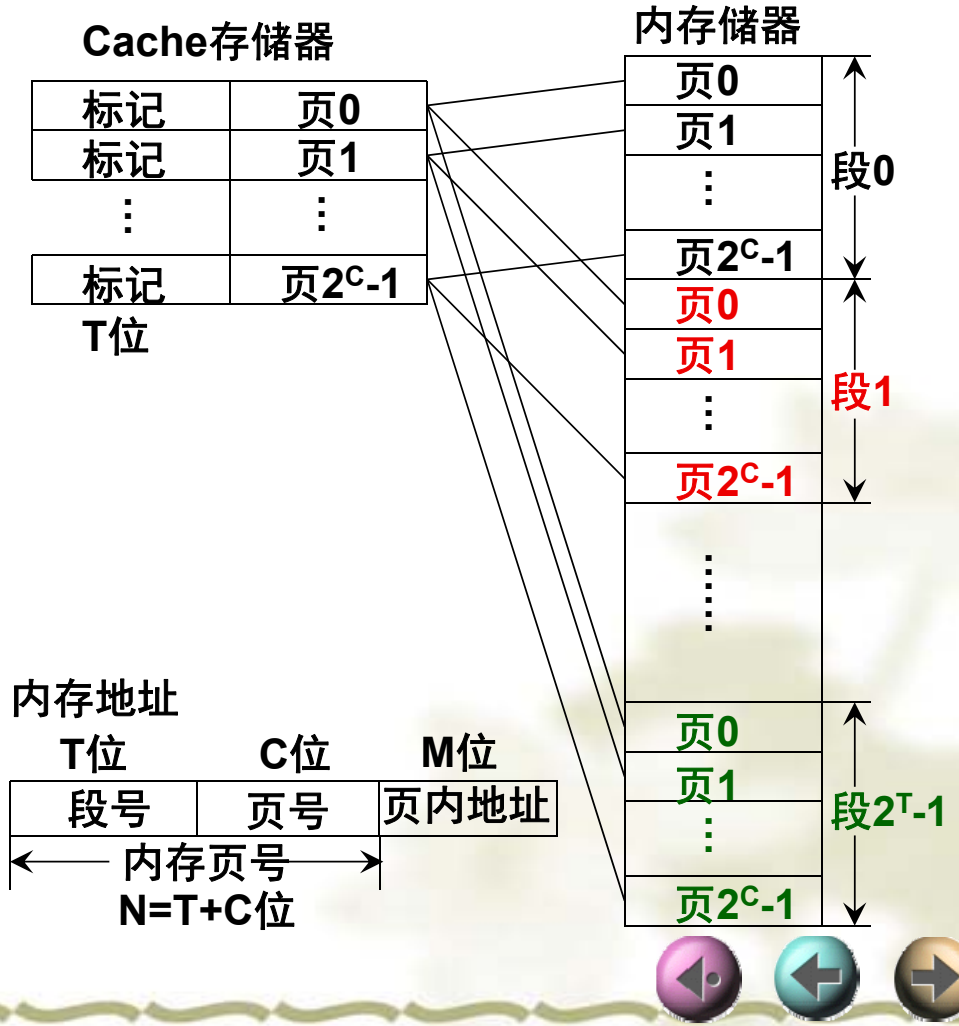
Cache和内存均分为若干个字节数相同的页。*内存中的任一页都可被调入Cache的任一页中*，所调入页的页号需全部存入地址索引机构中。寻址时，需将寻址地址同索引机构中的全部标记地址(页号)进行比较。



2. 直接映射方式

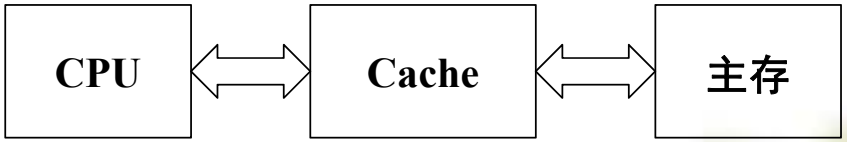
Cache中全部单元被划分成大小固定的页；内存则被划分成段，段再被划分成与Cache大小相同的页。

Cache中的各页只接收内存中相同页号的内容，地址索引机构中存放的标记地址是内存的段号。



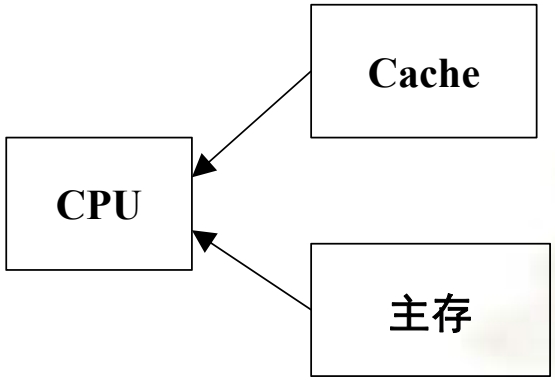
1. Cache的读过程

- 1. 贯穿读出式
  - CPU对主存的所有数据请求都首先送到Cache，在Cache中查找。
  - 若命中，切断CPU对主存的请求，并将数据送出；
  - 如果不命中，则将数据请求传给主存。



1. Cache的读过程

- 2. 旁路读出式
  - CPU向Cache和主存同时发出数据请求。
  - 命中，则Cache将数据回送给CPU，并同时中断CPU对主存的请求；
  - 若不命中，则Cache不做任何动作，由CPU直接访问主存

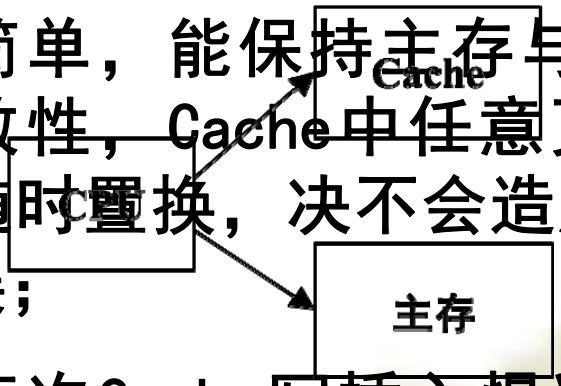


Cache的写过程

- 通写法
- 回写法
- 只写主存

(1) 通写(Write-Through)法

每次写入Cache时，同时也写入主存，使主存与Cache相关页内容始终保持一致。



- 优点：简单，能保持主存与Cache副本的一致性，Cache中任意页的内容都可被随时置换，决不会造成数据丢失的错误；
- 缺点：每次Cache写插入慢速的访主存操作，影响工作速度。

Cache的写过程

- 通写法
- 回写法
- 只写主存

(2) 回写法

每次只是暂时将数据写入Cache，并用标志将该页加以注明。

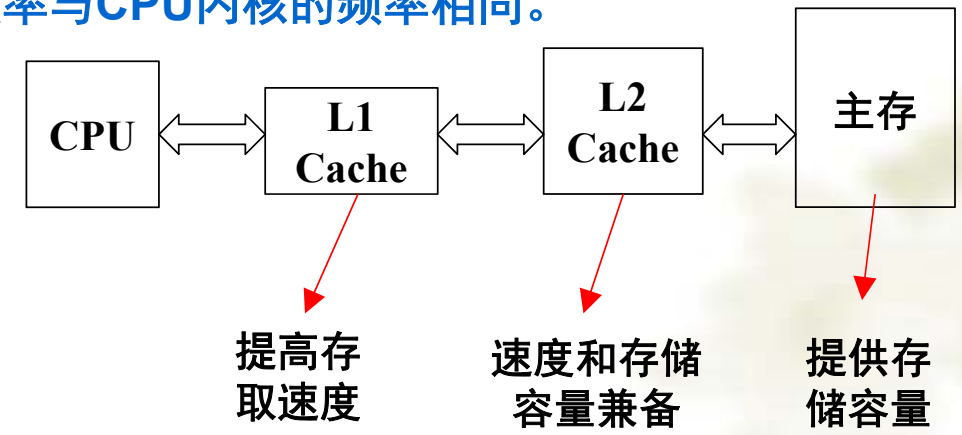
当Cache中任一页数据被置换时，只要在它存在期间发生过对它的写操作，那么在该页被覆盖之前必须将其内容写回到对应主存位置中去；

如果该页内容没有被改写，则其内容可以直接淘汰，不需回写。

这种方法的速度比通写法快，但结构要复杂的多，而且主存中的页未经随时修改，可能失效。

Cache的分级体系结构

- 一级Cache：容量一般为8KB~64KB
  - **一级Cahce集成在CPU片内。L1 Cache分为指令Cache和数据Cache。使指令和数据的访问互不影响。指令Cache用于存放预取的指令。数据Cache中存放指令的操作数。**
- 二级Cache：容量一般为128KB~2MB
  - **在Pentium II之后的微处理器芯片上都配置了二级Cache，其工作频率与CPU内核的频率相同。**



6.1.3 接口的基本功能

不同外设的接口，其功能及与外设的连接、通信方式各不相同。但任何接口电路的基本功能是相同的，有三：

- 作为微型机与外设传递数据的缓冲站；
- 正确寻址与微机交换数据；
- 提供微型机与外设间交换数据所需的读/写控制信号。

总之，就是完成三大总线的转换和连接任务。

6.1.4 接口的典型结构

1. 接口的基本结构

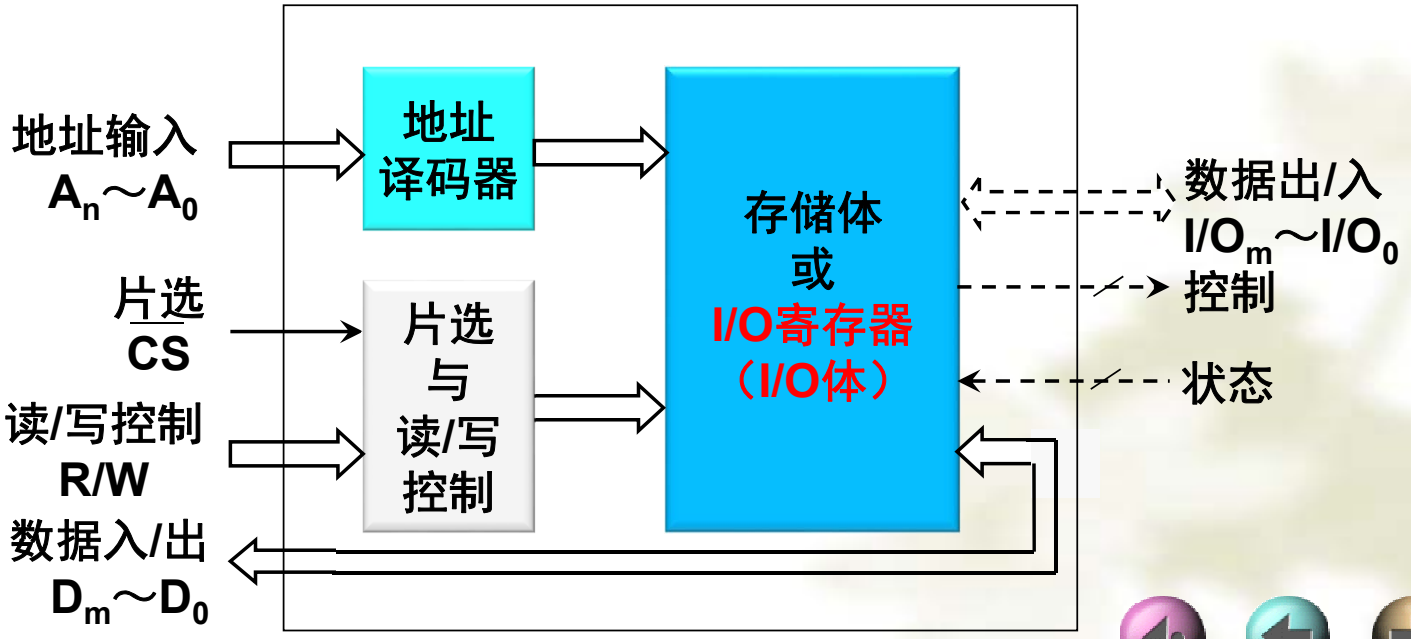
与接口的基本功能相对应，接口电路必须包含以下三种基本逻辑部件：

- I/O数据缓冲寄存器
- 寄存器地址译码器
- 读写控制逻辑

对于一些比较复杂的接口，为了增强功能和适应不同I/O同步控制方式的需要，往往还要引入一些别的逻辑电路。

6.2 I/O接口与存储器的本质共性（续）

③ 与CPU的外部连接特性相同。均有数据线、存储单元或I/O端口选择线、片选线和读/写控制线4类与CPU相连的外部引脚信号。

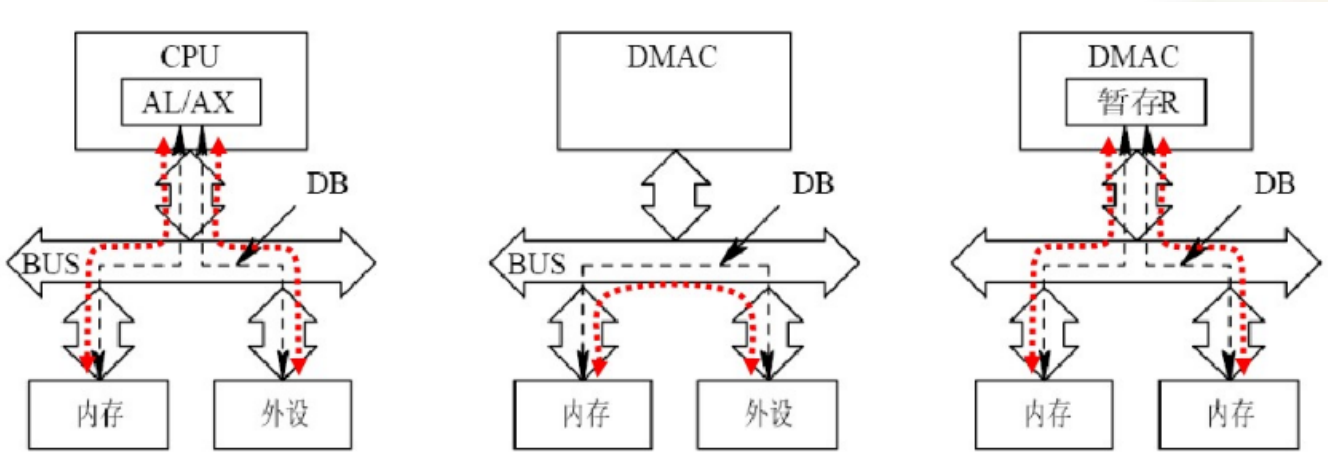


3. 常用I/O同步控制方式

I/O设备的同步控制方式通常有四种：

- *程序查询式控制*
- *中断驱动式控制*
- *直接存储器存取式控制*
- *延时等待式控制*

- DMA方式是一种由专门的硬件电路执行I/O交换的传送方式，它让外设接口与内存直接进行告诉的数据交换，而不必经过CPU，从而实现对存储器的直接存取，并获得总线控制权，来实现内存与外设或者内存的不同区域之间大量数据的快速传送。这种专门的硬件叫DMA控制器，简称DMAC。
- 特点
  - **外设直接与存储器进行数据交换，CPU不再担当数据传输的中介者**
  - **总线由DMA控制其（DMAC）进行控制（CPU要放弃总线控制权），内存/外设的地址和读写控制信号均由DMAC提供。**



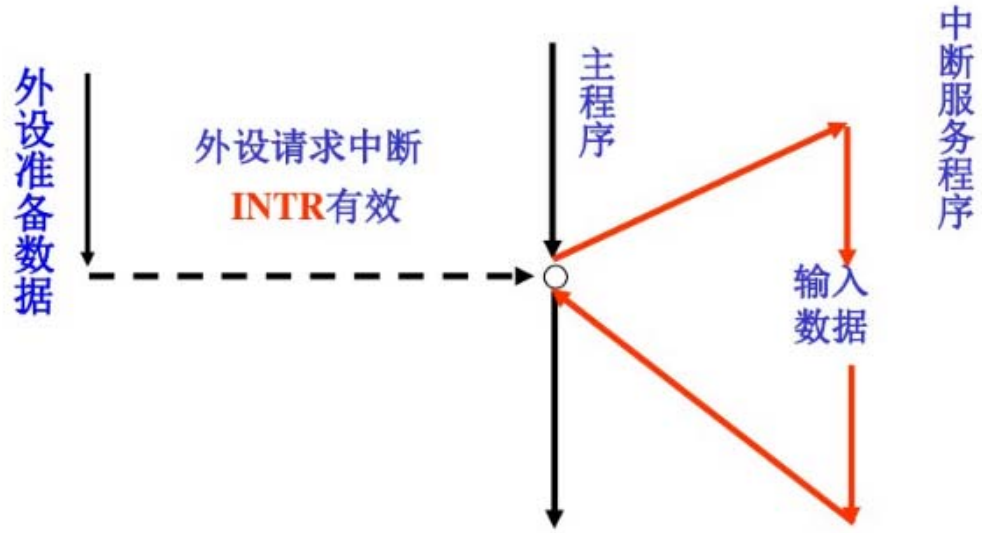
- 无条件传送
- 查询式传送

（程序控制输入/输出方式）

- 中断方式传送
- 直接存储器存取（DMA）

- 其中无条件方式和查询方式又称为程序控制方式，指用输入/输出指令，来控制信息传输的方式，是一种软件控制方式。

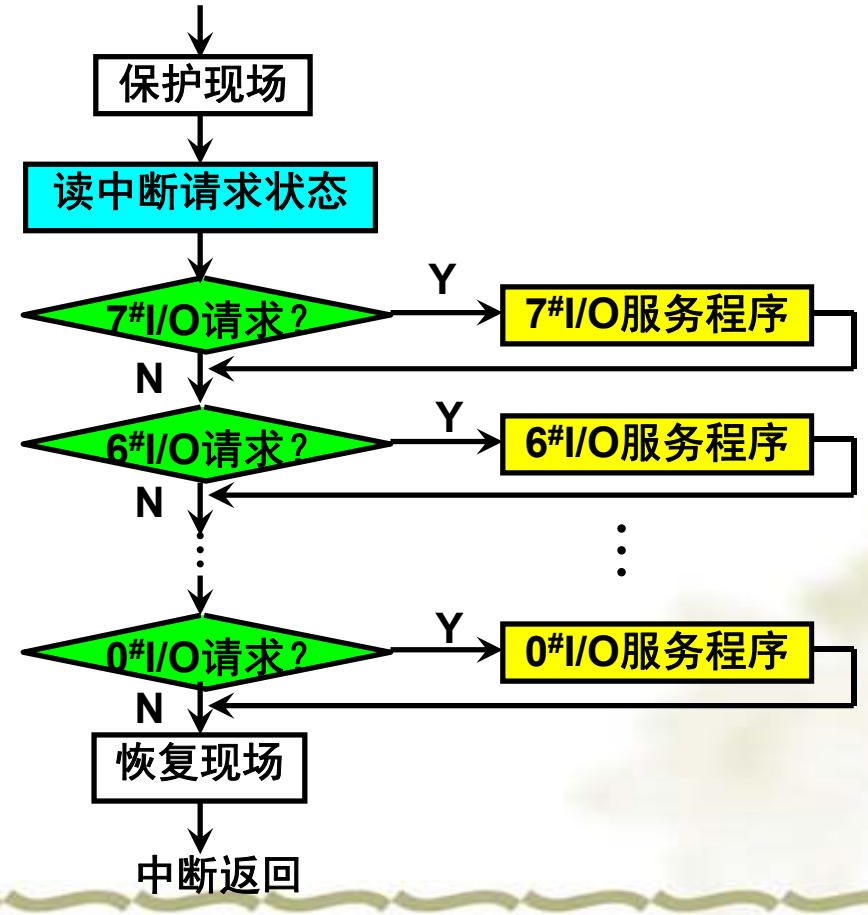
- 中断：
  - CPU执行程序时，由于发生了某种随机的事件（外部或内部），引起CPU暂时中断正在运行的程序，转去执行一段特殊的服务程序（称为中断服务程序或中断处理程序），以处理该事件，该事件处理完后又返回被中断的程序继续执行，这一过程称为中断。
  - 中断源：引起中断的事件，发出中断请求的来源。



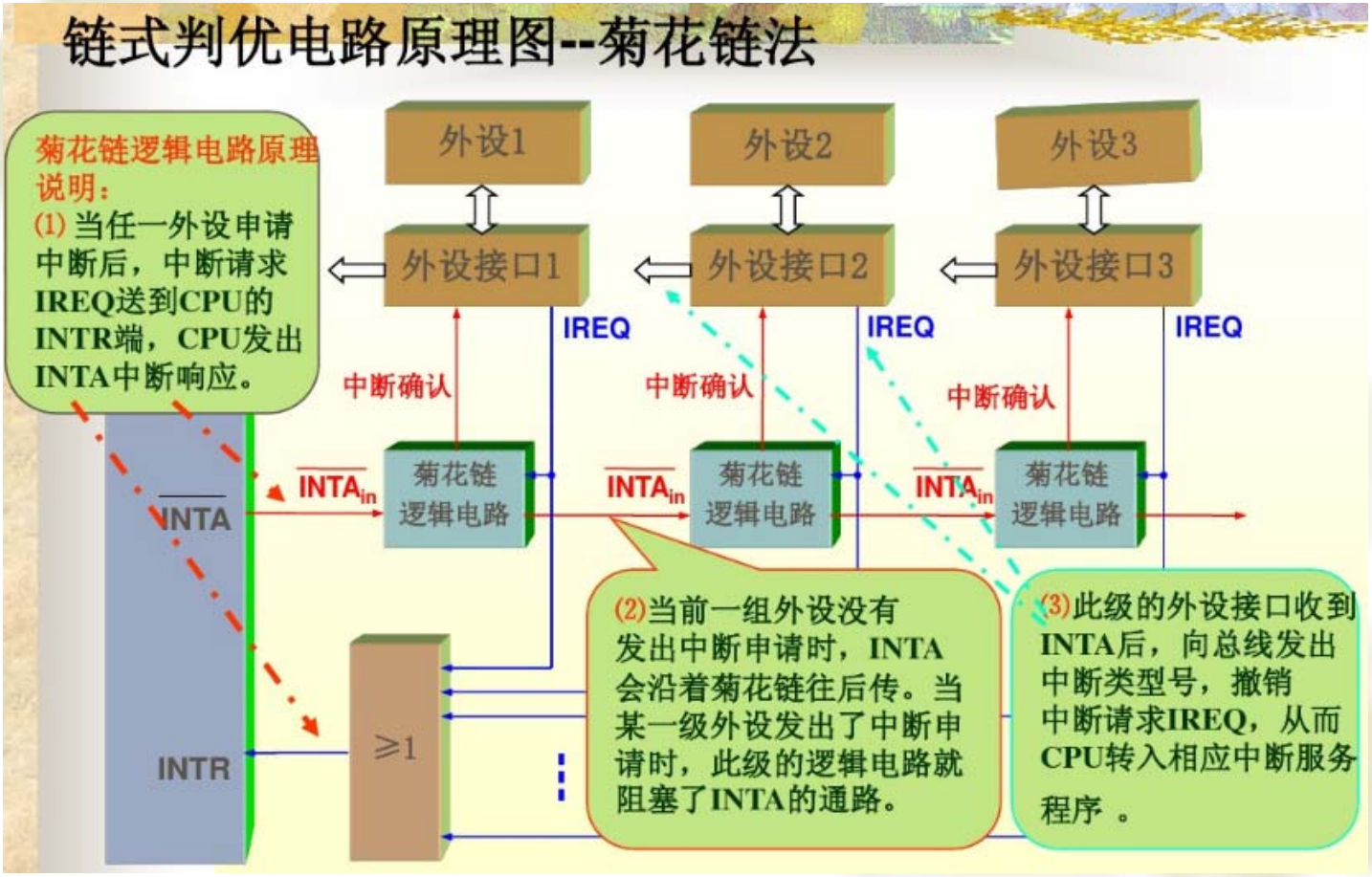
中断处理的一般过程

- 1. 中断请求
- 2. 中断源识别及中断判优
- 3. 中断响应
- 4. 中断处理（服务）
- 5. 中断返回

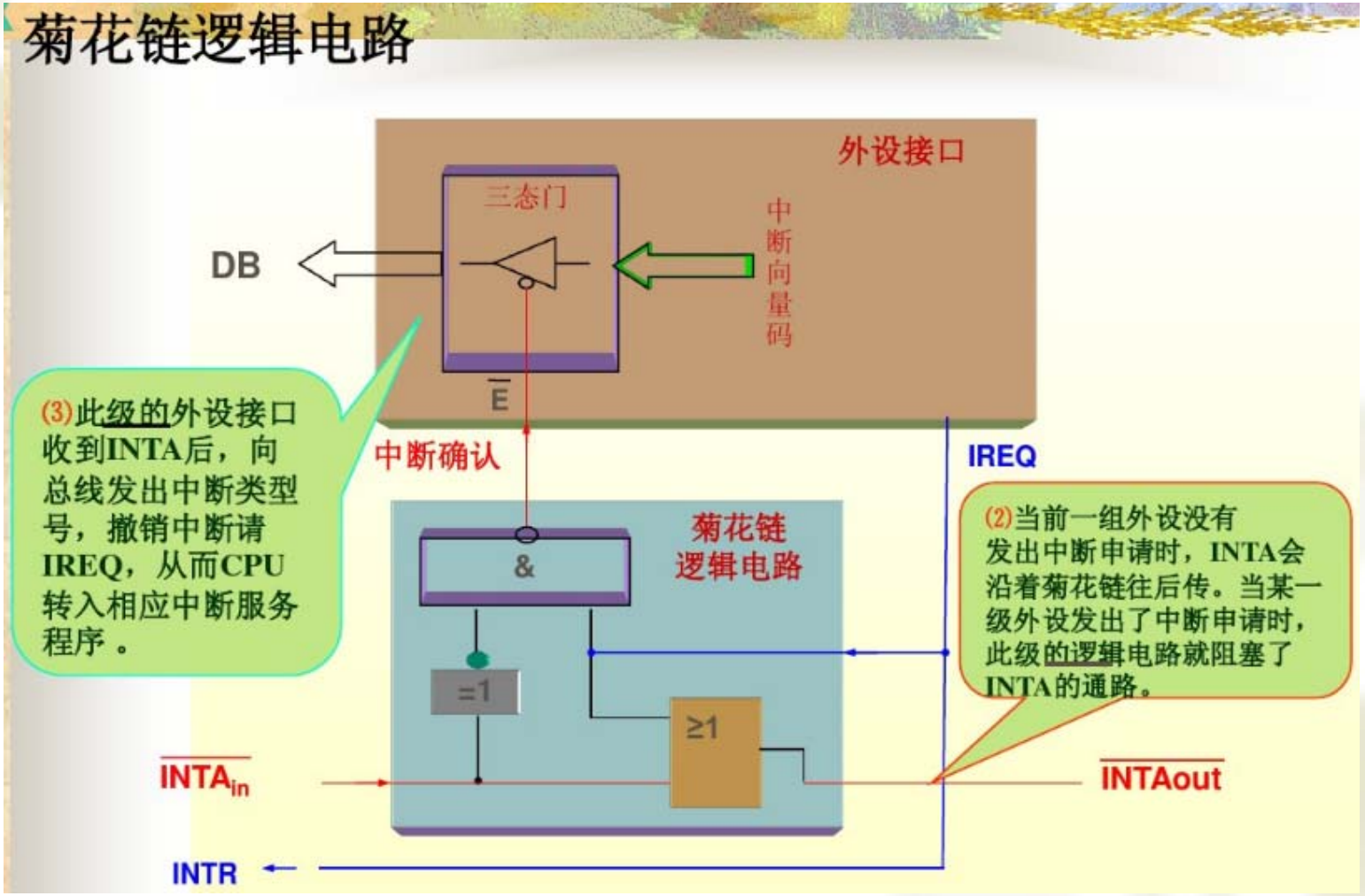
查询式中断处理程序流程图



链式判优电路原理图--菊花链法

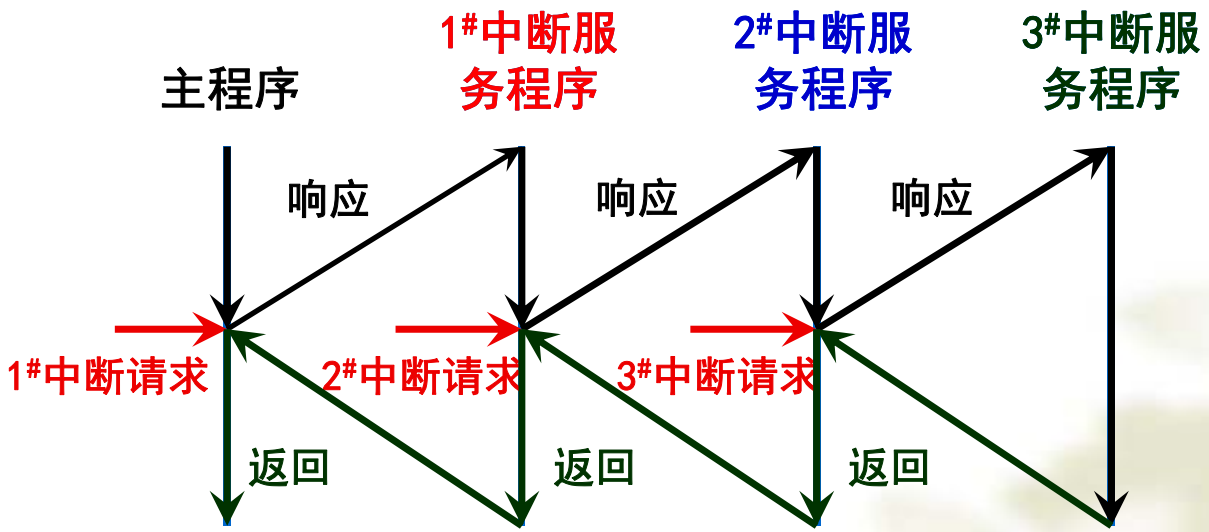


菊花链逻辑电路


外设接口
三态门
中断向量码
DB
$\overline{E}$
中断确认
IREQ
菊花链逻辑电路
&
=1
≥1
$\overline{INTA_{in}}$
$\overline{INTAout}$
INTR
(3)此级的外设接口收到INTA后，向总线发出中断类型号，撤销中断请IREQ，从而CPU转入相应中断服务程序 。
(2)当前一组外设没有发出中断申请时，INTA会沿着菊花链往后传。当某一级外设发出了中断申请时，此级的逻辑电路就阻塞了INTA的通路。

3. 中断嵌套示意

中断优先级：3#＞2#＞1#



嵌套的级数原则上不限，只取决于堆栈深度，实际上与要求的中断响应速度也有关。

CPU响应外部可屏蔽中断请求的条件

- 置位了中断请求触发器。
- 中断屏蔽触发器处于非屏蔽状态。
- CPU内部是中断开放的（CPU内部中断允许触发器IF=1）。
- 没有更高优先级别的中断请求正在被响应或正发出、正挂起。
- CPU正在执行的现行指令已经结束。

3. 中断响应

- 向中断源发出$\overline{INTA}$中断响应信号
- 保护硬件现场
  - 将FLAGS压入堆栈
- 保护断点
  - 将CS、IP压入堆栈
- 获得中断服务程序入口地址

由硬件系统完成

标志寄存器（FLAGS / PSW）

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				OF	DF	IF	TF	SF	ZF		AF		PF		CF

条件码标志：

OF 溢出标志
SF 符号标志
ZF 零标志
CF 进位标志
AF 辅助进位标志
PF 奇偶标志

控制标志：

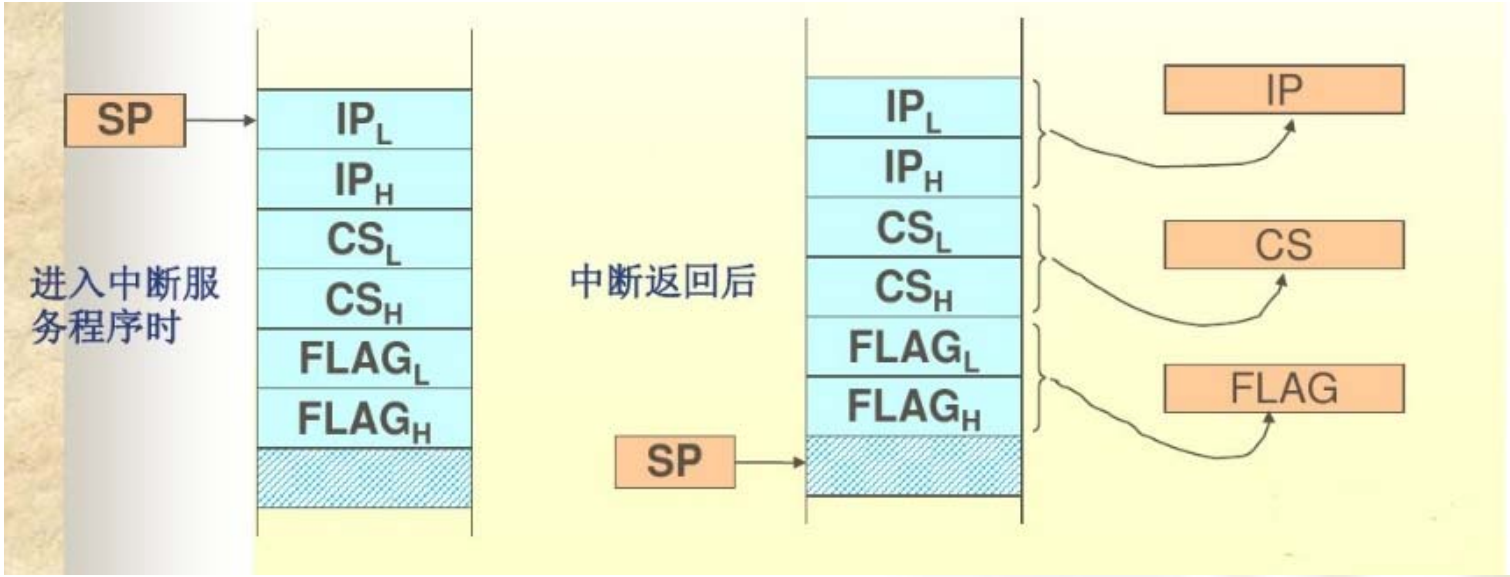
DF 方向标志
IF 中断标志
TF 陷阱标志

例：ADD AX, BX
JO / JC ERROR ?

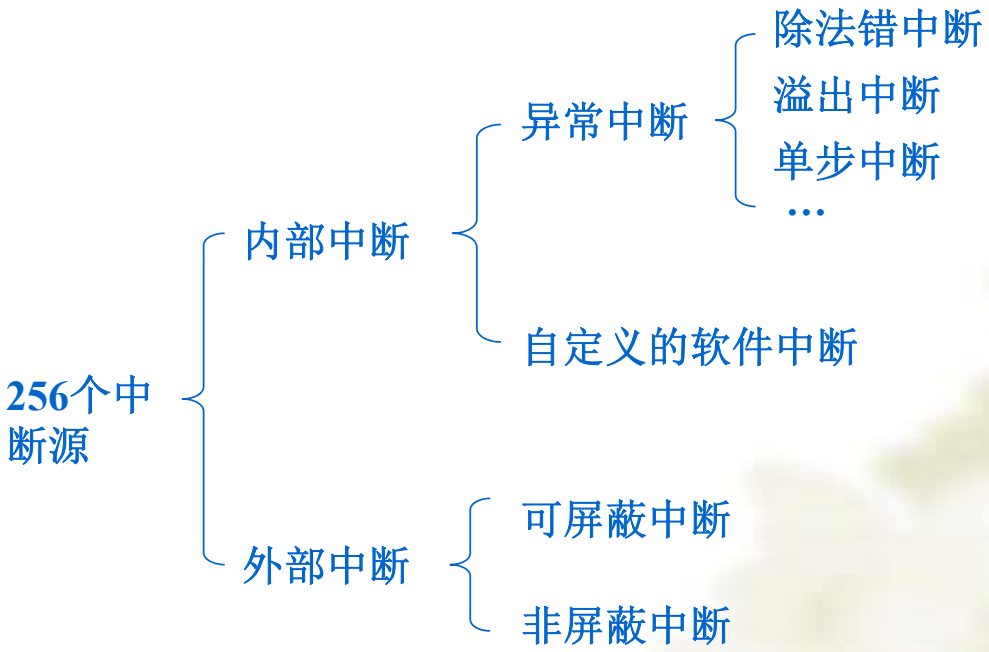
- 4. 中断处理
  - 执行中断服务子程序
  - 中断服务子程序的特点
    - 为“远过程”
    - 用IRET指令返回
  - 中断服务子程序完成的工作
    - 保护软件现场（参数）
    - 开中断（STI）
    - 中断处理
    - 关中断（CLI）
    - 恢复现场
    - 中断返回

- **5. 中断返回**
  - 执行IRET指令，使IP、CS和FLAGS从堆栈弹出

恢复断点和硬件现场



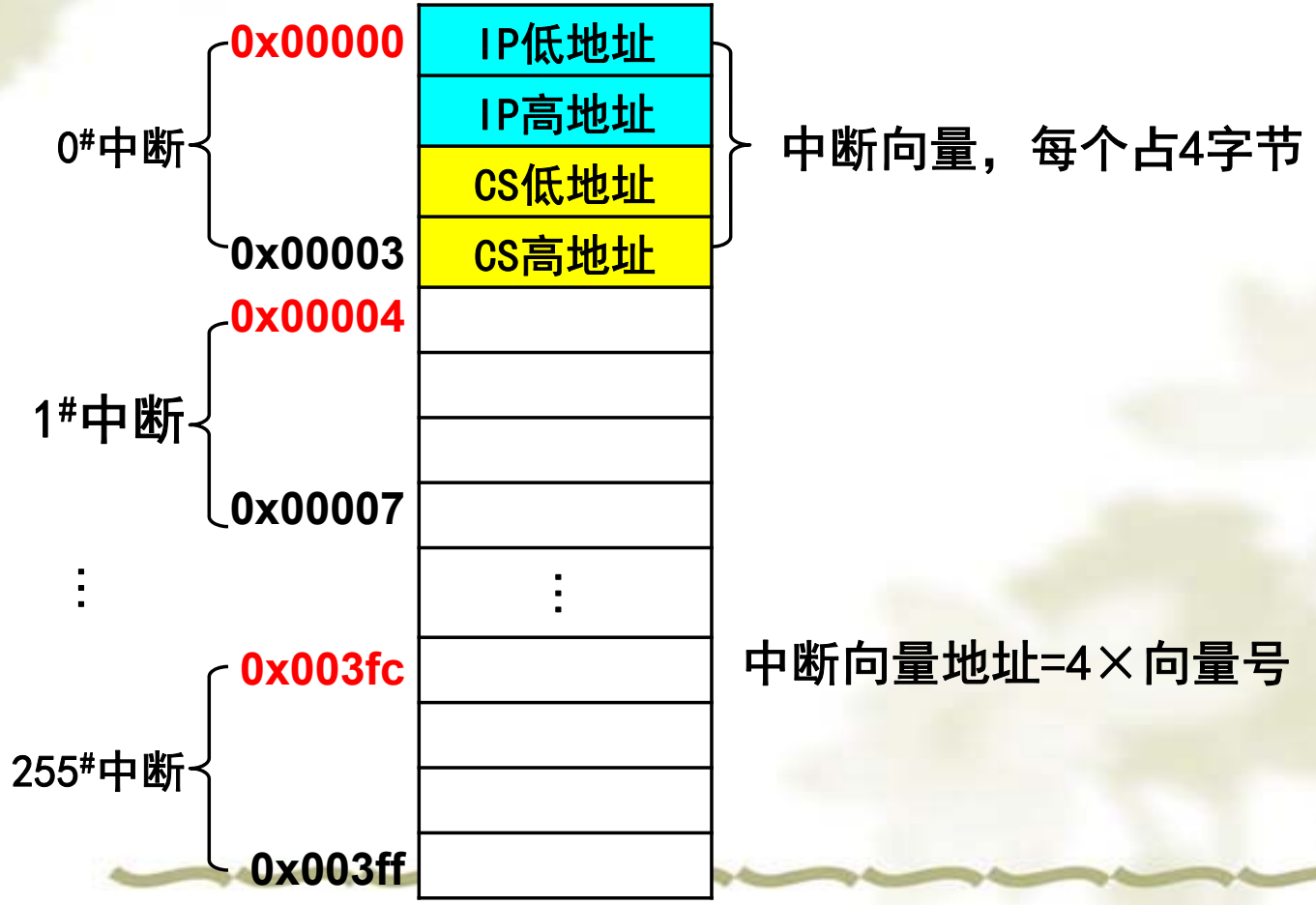
- 8086/8088为每个中断源分配 一个中断类型码（中断向量码），其取值范围为0~255，实际可处理56种中断。其中包括软件中断，系统占用的中断，已经开放给用户使用的中断。所有中断又可分为两大类：内部中断和外部中断。



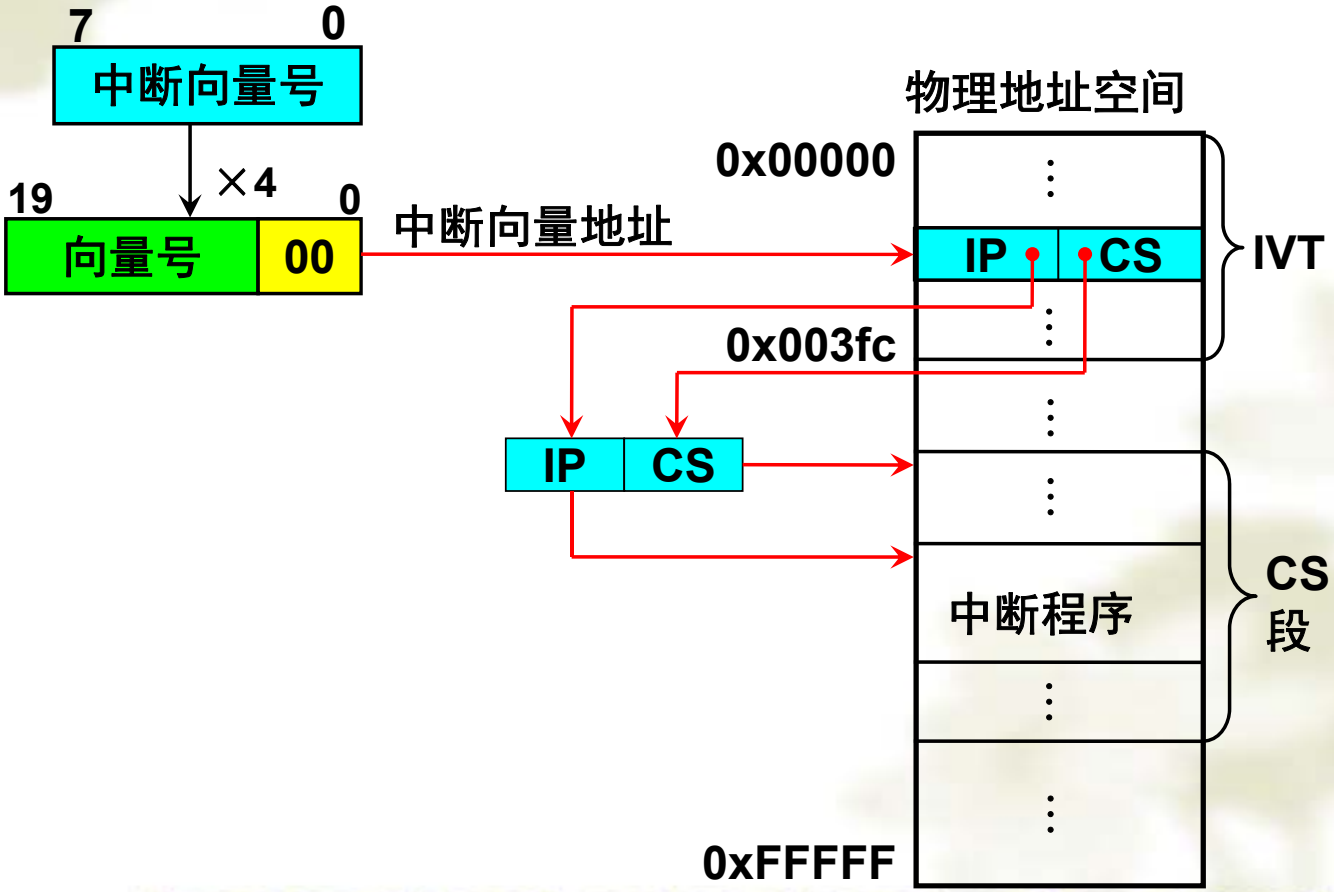
- 1. 内部中断源
  - 内部中断源又被称为软件中断，即根据某条指令或者对标志寄存器中某个标志的设置而产生，它与硬件电路无关。
  - （1）除法出错中断——0型中断，除数为0或商超过了结果寄存器所能表示的最大范围
  - （2）单步中断——1型中断，标志寄存器中有一位陷阱标志TF。
  - （3）断点中断——3型中断，专用于设置断点的指令INT 3，用于程序中设置断点来调试程序。
  - （4）溢出中断——4型中断，在算数指令的执行过程发出溢出
  - （5）用户自定义的软件中断——n型中断，执行中断指令INT n引起内部中断。

- 2. 外部中断源（硬件中断）
  - 由外部的硬件或外设接口产生的中断。
  - （1）不可屏蔽中断：由NMI引脚引入，上升沿触发，不受中断允许标志IF的影响，每个系统中仅允许有一个，都是用来处理紧急情况的，如掉电处理。这种中断一旦发生，系统会立即响应。
  - （2）可屏蔽中断：由INTR引脚引入，它受中断允许标志的影响，也就是说，只有当IF=1时，可屏蔽中断才能进入，反之则不允许进入，可屏蔽中断可有多个，一般是通过优先级排队，从多个中断源中选出一个进行处理。

中断向量表结构示意


0x00000
IP低地址
IP高地址
CS低地址
CS高地址
0x00003
0#中断
中断向量，每个占4字节
0x00004
1#中断
0x00007
0x003fc
255#中断
0x003ff
中断向量地址=4×向量号

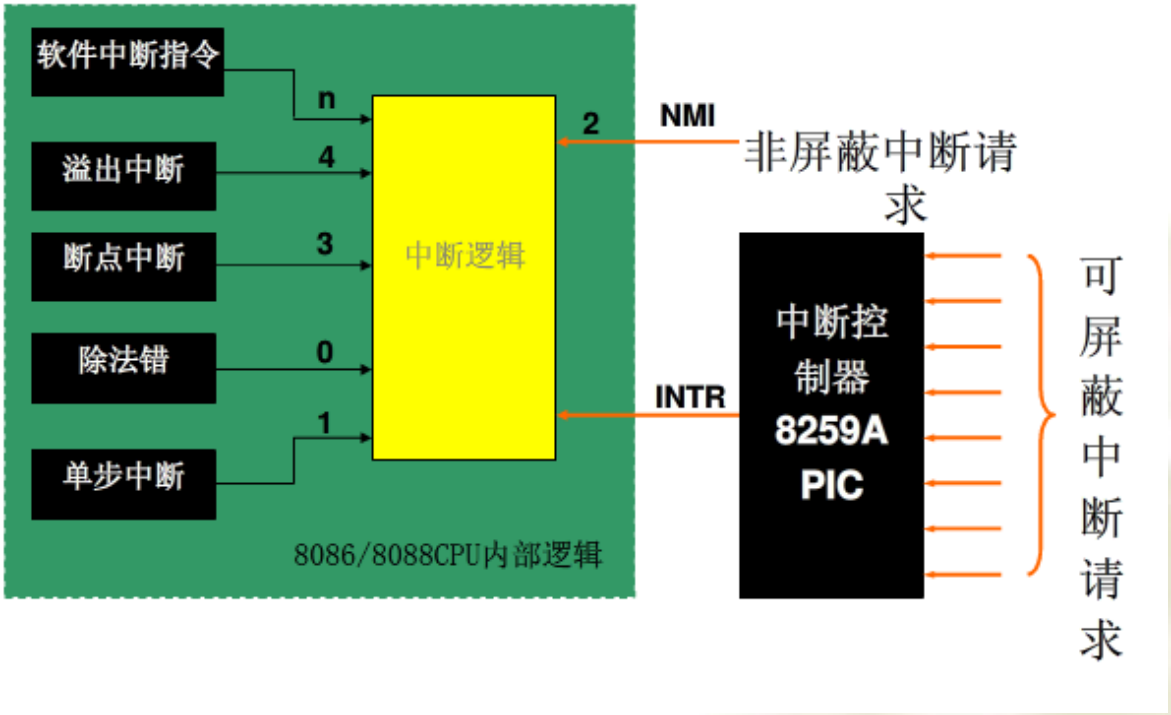
中断/异常处理程序的进入过程



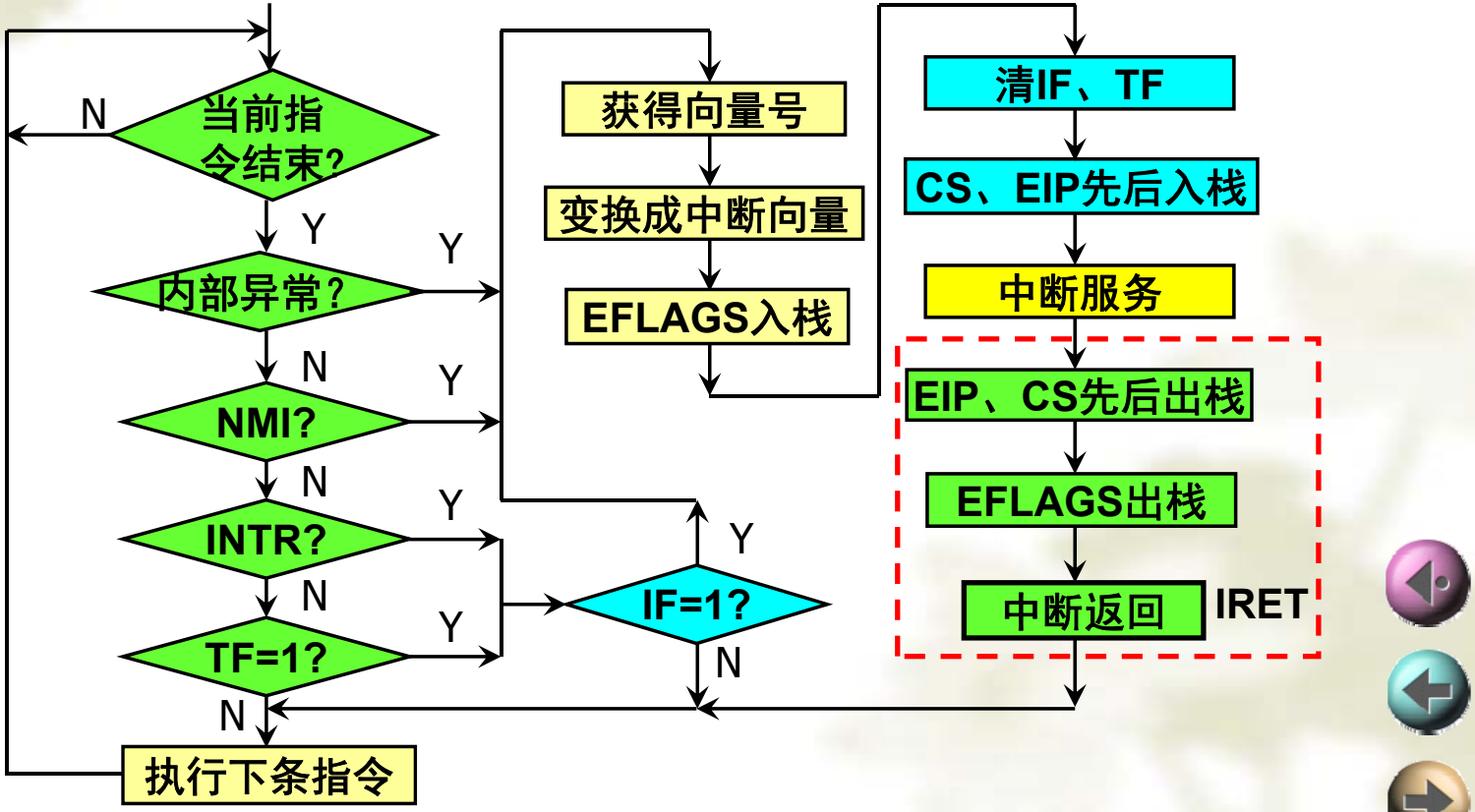
- 4. 8086/8088 CPU的中断响应过程
  - （1）内部中断响应过程。
    - ◆对于除法溢出、单步、断点和溢出中断，中断类型码自动形成，INT n指令则是直接由n指出。
    - ◆（1）乘以4得到中断向量的地址
    - ◆（2）硬件保护现场，标志寄存器FLAGS压入堆栈。
    - ◆（3）清除IF和TF标志，屏蔽新的INTR中断和单步中断。
    - ◆（4）保存断点
    - ◆（5）将中断服务子程序的入口地址分别送至CS和IP内。
    - ◆（6）转去中断服务子程序执行。

- （2）外部中断响应过程。
  - 非屏蔽中断响应：不用外部接口给出中断类型码，CPU会自动按中断类型码2来计算中断向量的地址。处理过程和内部中断一样。
  - 可屏蔽中断响应：如果中断允许标志IF=1，则CPU会在当前指令执行完毕后，产生两个连续的中断响应总线周期。
    - 第一个总线周期
      - 地址/数据总线置高阻
      - 发出第一个中断响应信号/INTA给中断控制器
      - 启动LOCK信号，通知总线仲裁器8289，使系统其他处理器不能访问总线。
    - 第二个总线周期
      - CPU送出第二个/INTA信号，通知中断控制器将相应中断请求的中断类型码放到数据总线上供CPU读取。

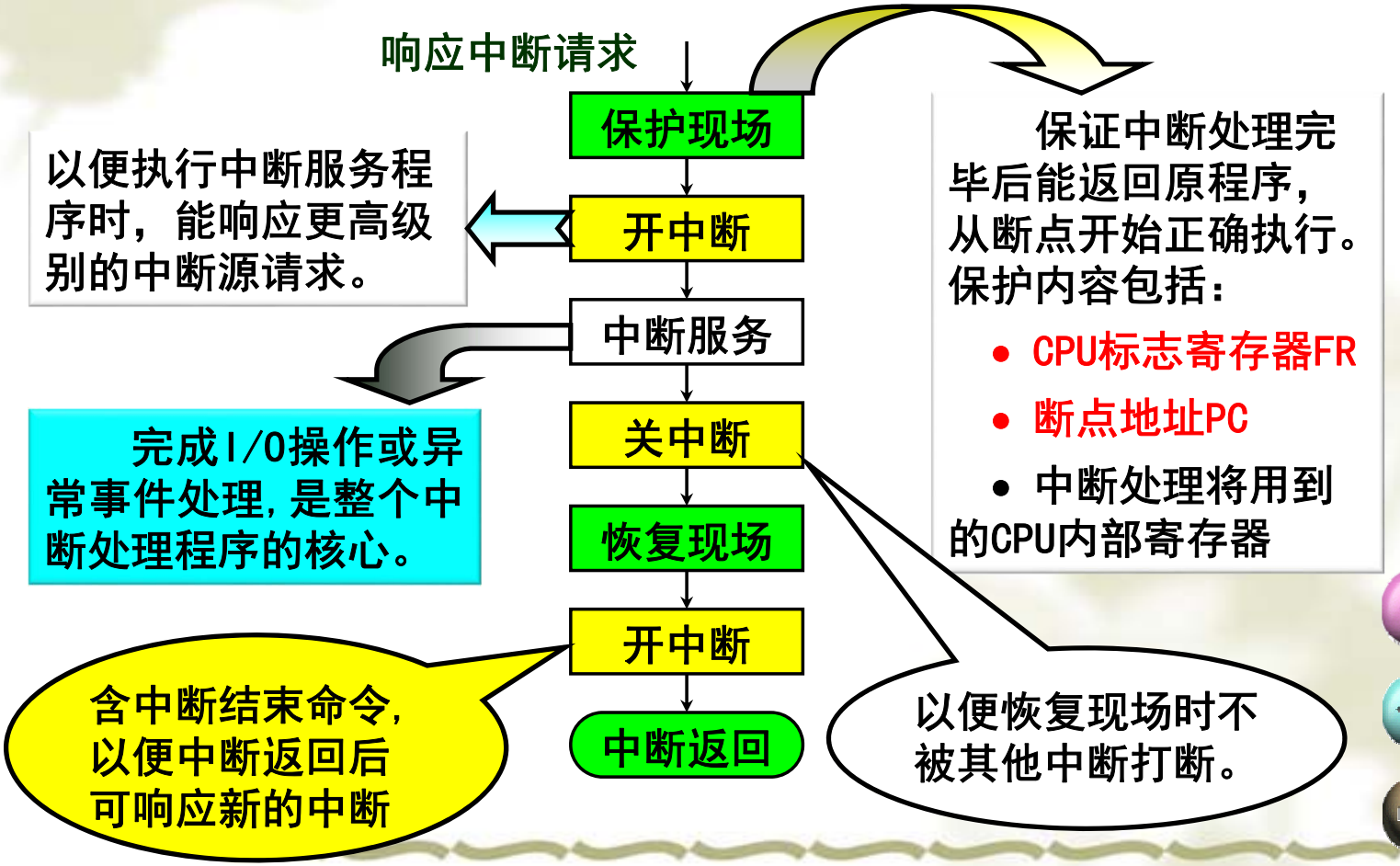
- 8088/8086中断系统



6.2.4 中断/异常的检测、响应、处理过程



6.4.2 中断处理子程序的设计



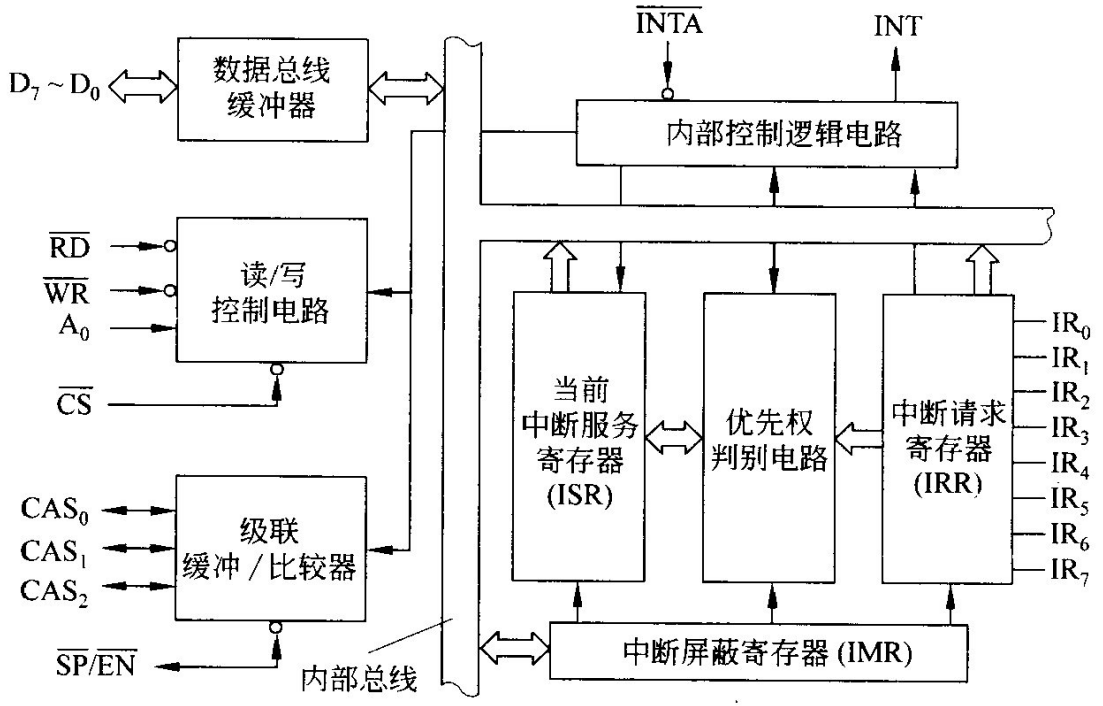


图 6-25　8259A 内部结构框图

8259A的初始化编程

- **1. 8259A内部寄存器的寻址方式**

表 6-1　8259A 内部寄存器的访问方法

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A_0	D_4	D_3	读写操作
0	1	0	0	0	0	写入 OCW2
			0	0	1	写入 OCW3
			0	1	×	写入 ICW1
			1	×	×	写入 ICW2、ICW3、ICW4、OCW1(顺序写入)
0	0	1	0	—	—	读出 IRR、ISR
			1	—	—	读出 IMR

- **8259 实际端口地址**

  一线二址，址1：$A_0 = 0$；址2：$A_0 = 1$

- **8259 应操作的端口**

  4 个初始化命令字的写操作

  3 个操作命令字的写操作

  3 个寄存器 IRR、ISR、IMR 的读操作

  1 个中断类型号的读操作

对8259的初始化一定要按规定的顺序进行，假定8259占用的**I/O地址**为FF00和FF02H(奇地址)：

```
MOV DX, 0FF00H;          8259的地址A0=0
MOV AL, 13H;             写ICW1，边沿触发，单片，10011
OUT DX, AL
MOV DX, 0FF02H;          8259地址A0=1
MOV AL, 48H;             写ICW2，设置中断类型码
OUT DX, AL;              中断向量为48H-4FH（IR0-IR7）单片8259，不对
ICW3设置
MOV AL, 03H;             写ICW4，8086/88模式，自动中断结束，非缓冲，
一般嵌套，00000011
OUT DX, AL
MOV AL, 0E0H;            写OCW1，屏蔽IR5、IR6、 IR7中断源，11100000
OUT DX, AL;              (假定这3个中断输入未用)，其它开中断
```

- 2. 内部结构和工作原理
  - 1）计数器（0、1、2）
  - 2）控制寄存器
  - 3）数据总线缓冲器
  - 4）读写控制逻辑

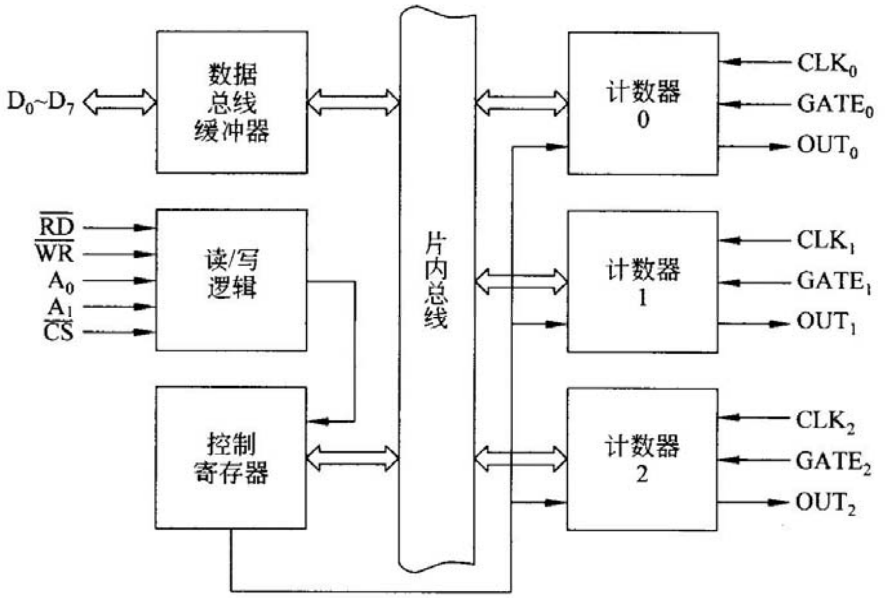


图 7-5　可编程定时器 8253 的内部结构框图

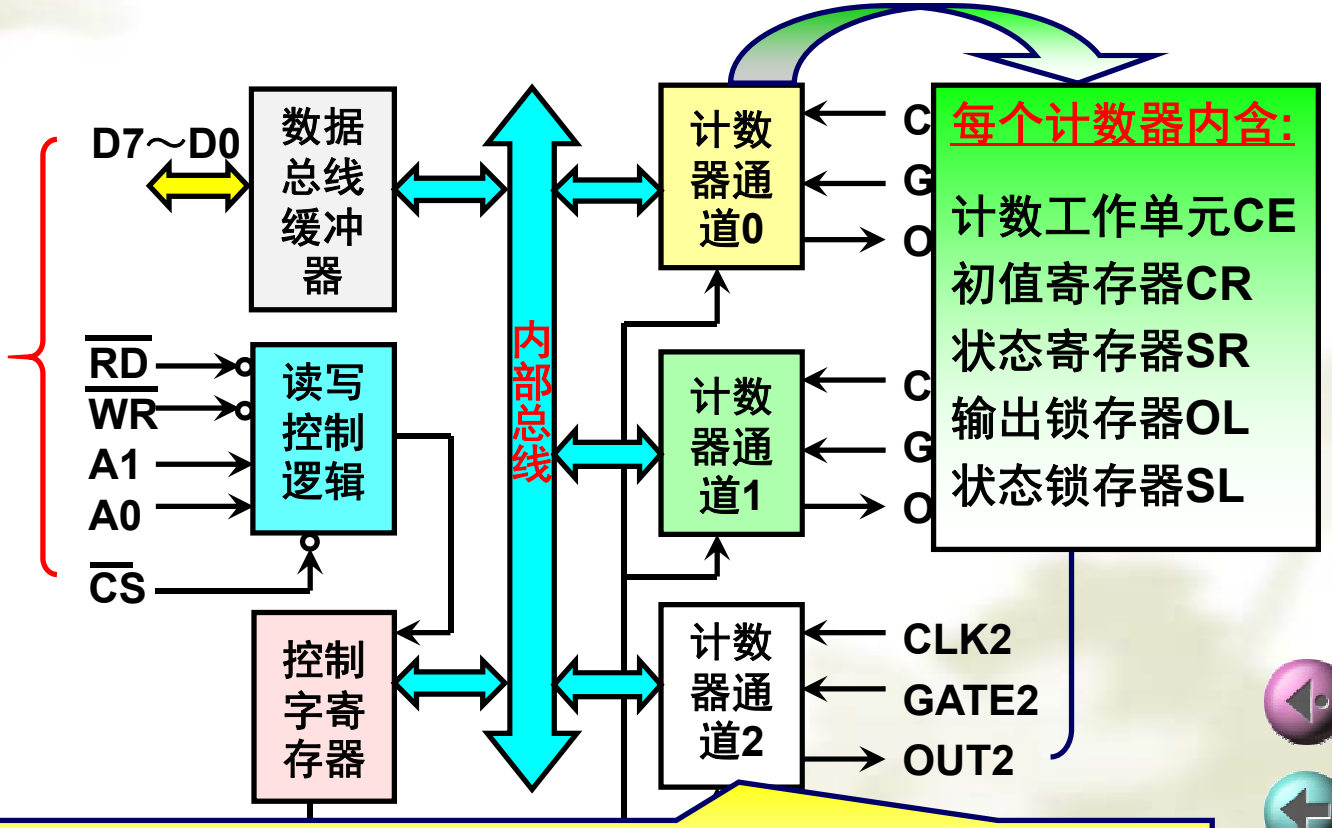
7.2.1 8254的基本功能

8254具有以下基本功能：

- 有3个独立的16位计数器通道
- 每个计数器可按二进制或十进制(BCD)计数
- 每个计数器可工作于6种不同工作方式
- 每个计数器允许的最高计数频率为10MHz (8253为2MHz，8253-5为5MHz)
- 有读回命令(8253没有)，可以读出当前计数单元的内容和状态寄存器内容

7.2.2 内部结构与外部引脚

与微机相连的引脚，接口特性与8259和各类存储器芯片类似



每个计数器既可用作计数器，又可用作定时器，差别在于：*计数脉冲间隔不一定相同，而定时脉冲要求周期一定。*

7.2.3 内部端口寻址与读写控制

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A1	A0	读/写操作说明
0	1	0	0	0	写计数通道0的CR
0	1	0	0	1	写计数通道1的CR
0	1	0	1	0	写计数通道2的CR
0	1	0	1	1	写控制寄存器
0	0	1	0	0	读通道0的OL或状态锁存器
0	0	1	0	1	读通道1的OL或状态锁存器
0	0	1	1	0	读通道2的OL或状态锁存器
0	0	1	1	1	无操作
1	×	×	×	×	禁止使用
0	1	1	×	×	无操作

7.2.4 六种工作方式

8254各计数器通道均有6种工作方式可供选择：

1. 方式0 — 计数结束中断方式
2. 方式1 — 硬件可重触发单稳方式
3. 方式2 — 速率波发生器方式
4. 方式3 — 方波方式
5. 方式4 — 软件触发选通方式
6. 方式5 — 硬件触发选通方式

计数器工作方式一览表

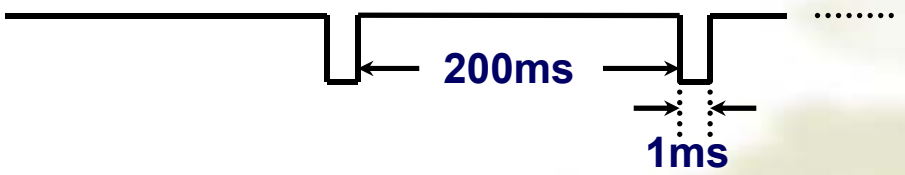
表 7-2　8253 计数器工作方式一览表

工作方式	启动计数	中止计数	自动重复	更新初值	输 出 波 形
0	软件	GATE=0	否	立即有效	延时时间可变的上升沿
1	硬件	/	否	下一轮有效	宽度为 $N\times T_{CLK}$ 的单一负脉冲
2	软/硬件	GATE=0	是	下一轮有效	周期为 $N\times T_{CLK}$，宽度为 T_{CLK} 的连续负脉冲
3	软/硬件	GATE=0	是	下半轮有效	周期为 $N\times T_{CLK}$ 的连续方波
4	软件	GATE=0	否	立即有效	宽度为 T_{CLK} 的单一负脉冲
5	硬件	/	否	下一轮有效	宽度为 T_{CLK} 的单一负脉冲

3）初始化举例

例7.1已知8254各端口地址分别为200H、201H、202H和203H，CLK_0接有1MHz的时钟，OUT_0接通道1时钟输入CLK_1，现希望OUT_1输出一个高电平为200ms、低电平为1ms的连续信号，编写初始化程序。

解：OUT_1输出波形为：

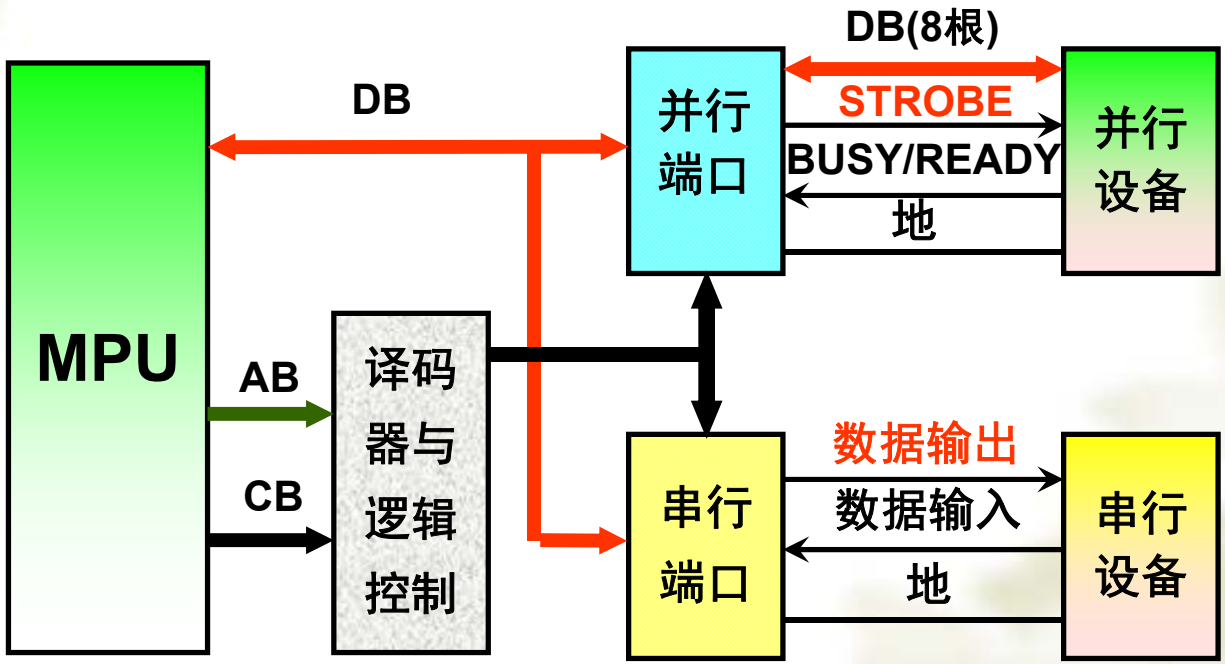


通道1工作于方式2，计数值=201ms/1ms=201

通道0要为通道1提供周期为1ms的时钟信号，所以应工作在方式2或方式3，计数初值为：

计数初值=1ms/1 μs=1000

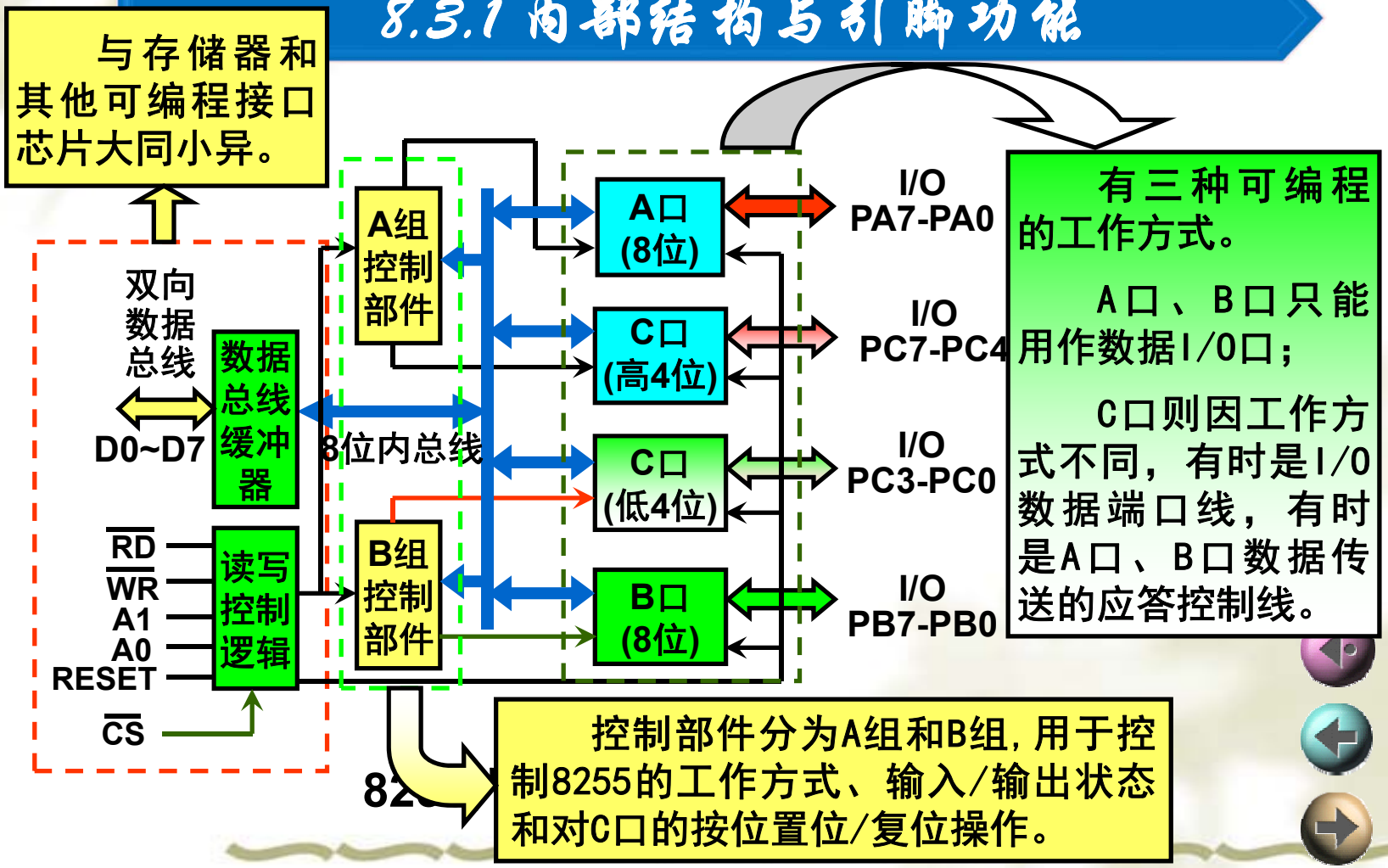
2. 并行与串行接口在结构、功能上的异同



主要差别：串行接口需要实行并行和串行之间的相互转化，而并行接口则无需实现这种变化。

8.3.1 内部结构与引脚功能

与存储器和其他可编程接口芯片大同小异。



有三种可编程的工作方式。

A口、B口只能用作数据I/O口；

C口则因工作方式不同，有时是I/O数据端口线，有时是A口、B口数据传送的应答控制线。

控制部件分为A组和B组，用于控制8255的工作方式、输入/输出状态和对C口的按位置位/复位操作。

8.3.2 内部端口寻址与读/写控制

8255内部共有A口、B口、C口和控制口4个端口寄存器,对他们的寻址和读/写操作是由CS^*、A1、A0和RD^*、WR^*几个信号来控制的。

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	操　作	
0	0	0	1	0	A口→数据总线	输入
0	1	0	1	0	B口→数据总线	
1	0	0	1	0	C口→数据总线	
0	0	1	0	0	数据总线→ A口	输出
0	1	1	0	0	数据总线→ B口	
1	0	1	0	0	数据总线→ C口	
1	1	1	0	0	数据总线→控制寄存器	
×	×	×	×	1	端口输出为”高阻”	禁止
1	1	0	1	0	非法	
×	×	1	1	0	端口输出为”高阻”	

- 1）A口、B口均为输入：
  - 利用C口的6条线作为选通控制信号线。A口使用PC3、PC4和PC5，而B口依然使用PC0、PC1和PC2。

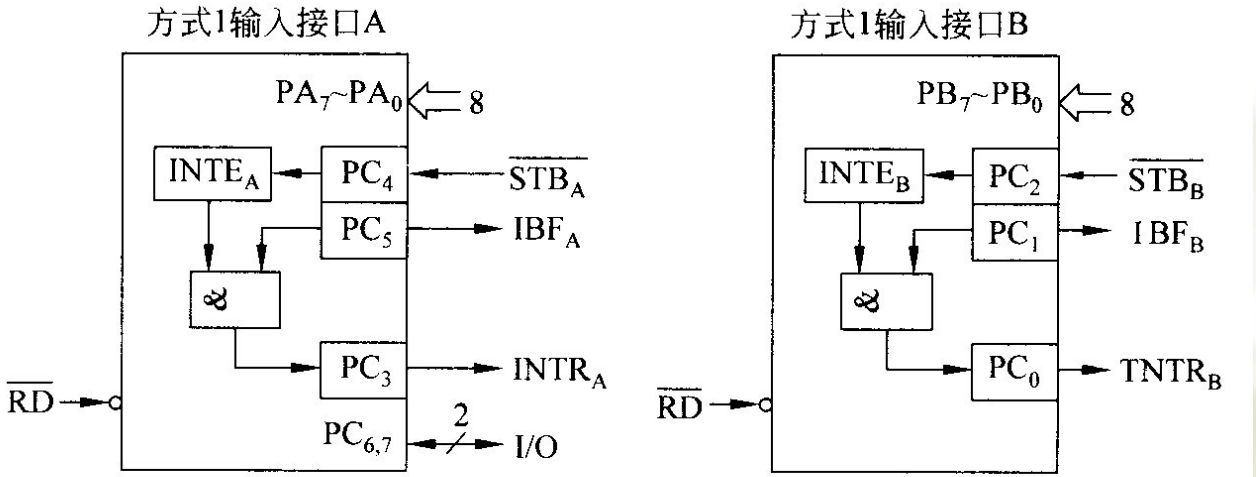


图 7-23　方式 1 下 A、B 口均为输入时的信号定义

8.7.1 INS8250的基本功能

① 包含发送控制电路和接收控制电路，可实现全双工通信。

② 可通过编程指定异步串行通信的数据格式。

③ 可通过编程设置串行数据传送波特率，并据此自动产生工作时钟输出。

④ 具有完整的状态报告功能。

⑤ 具有控制Modem的功能。

⑥ 通过设置内部LOOP控制位可提供循环反馈能力，实现数据自发自收功能

1. INS8250的控制字和状态字

- 线路控制寄存器(LCR)
- 线路状态寄存器(LSR)
- 除数寄存器(DLR)
- 中断允许寄存器(IER)
- 中断标识寄存器(IIR)
- MODEM控制寄存器(MCR)
- MODEM状态寄存器(MSR)

(3) 除数寄存器(DLR，+0/+1)

DLR是一个16位的寄存器，可分成高8位和低8位进行读/写操作，用于控制串行数据传送的波特率。除数寄存器与波特率之间的关系为：

除数寄存器值=基准时钟频率÷(16×波特率)

所以，在输入基准时钟频率确定后，可以通过改变除数寄存器的值来选择所需的波特率。

2. INS8250的初始化编程

初始化编程是指写入相应的控制字以规定8250的工作方式和通信参数等。