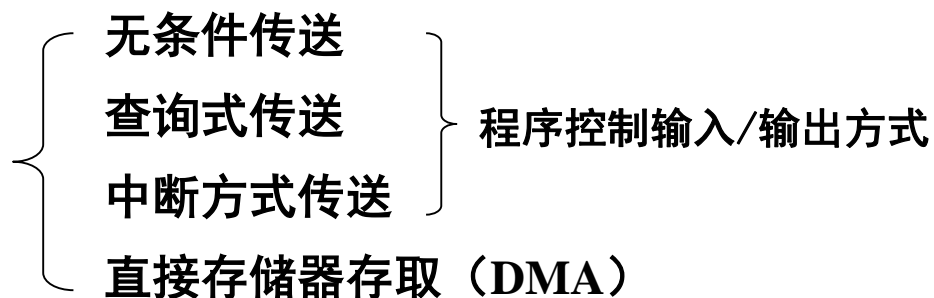


## 6.3 基本输入/输出方式



- 其中无条件方式、查询方式和中断方式又称为程序控制输入/输出方式，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。

## 6.3.1 无条件传送方式

- 利用程控方式与外设交换信息时，如果输入/输出的时刻，都可以保证外设总是处于“准备好”状态，则可以直接利用输入/输出指令进行信息的输入/输出操作。
- 应用场合：对外设的状态是可以预先确定的
- 优点：软件及接口硬件简单
- 缺点：只适用于简单外设，适应范围较窄
- 例子
  - 读取开关的状态
  - 当开关闭合时，输出编码使发光二极管亮

- 例子1，开关，我们想要知道开关K的当前状态。

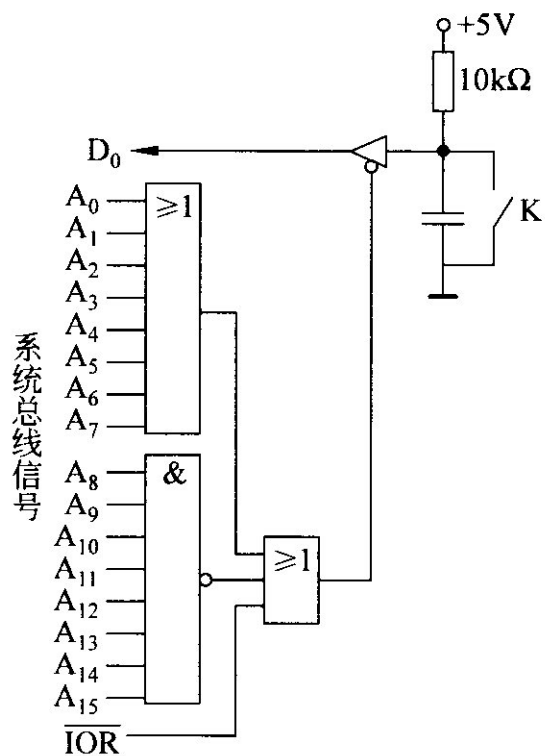


图 6-12 开关 K 通过三态门接口与系统的连接

- 例子2，发光二极管，我们想要让Q0和Q6端的发光二极管发光。

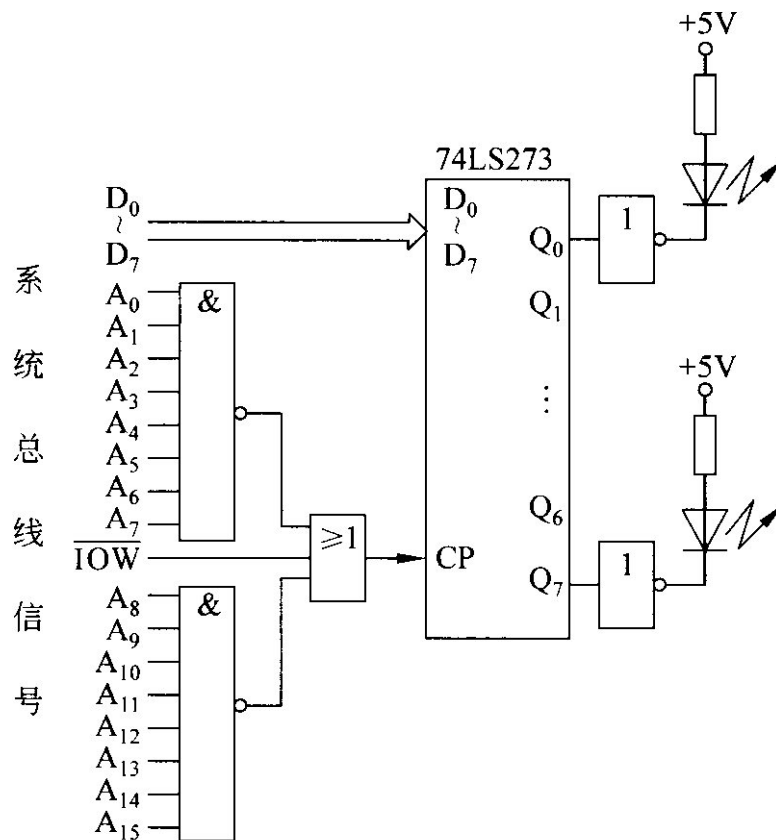


图 6-6 74LS273 作输出接口的应用

## 6.3.2 查询方式

- 通过程序查询相应设备的状态，若状态不符合，则CPU不能进行输入/输出操作，需要等待；只有当状态信号符合要求时，CPU才能进行相应的输入/输出操作。
- 一般外设均可以提供一些反映其状态的信号，如对输入设备来说，它能够提供“准备好” ready信号，ready=1表示输入数据已准备好。输出设备则提供“忙” busy信号，busy=1表示当前时刻不能接收CPU来的数据，只有当busy=0时，才表明它可以接收来自于CPU的输出数据。
- 仅当条件满足时才能进行数据传送
- 每满足一次条件只能进行一次数据传送
- 适用场合：
  - 外设并不总是准备好
  - 对传送速率和效率要求不高
- 工作条件：
  - 外设应提供设备状态信息
  - 接口应具备状态端口

## 6.3.2 查询方式

### ■ 查询方式流程图

- 对ready的状态查询，是通过读状态端口的相应位来实现的，输出的情况亦大致相同，这种传送控制方式的优点是，能够保证输入/输出数据的正确性。

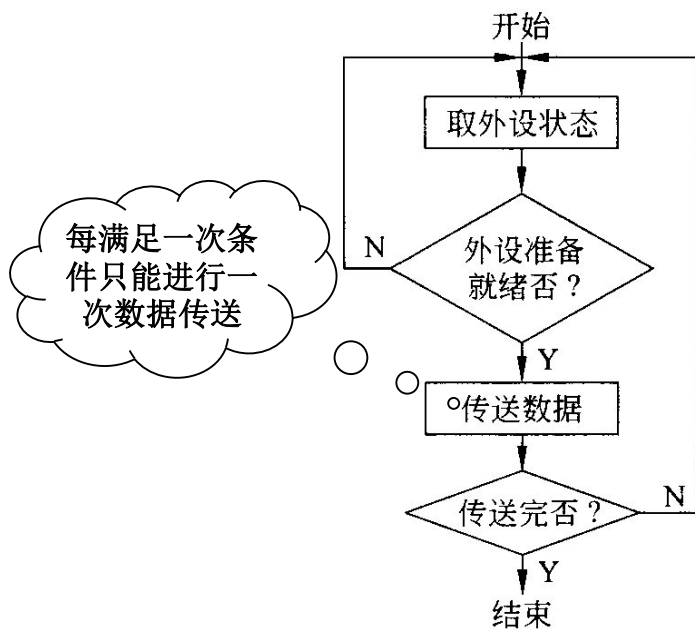


图 6-13 单一外设时的查询方式流程图

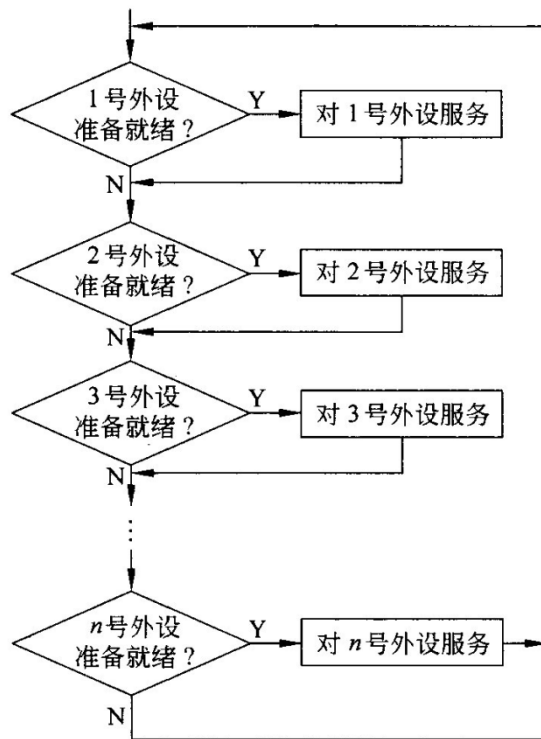


图 6-14 多个外设时的查询工作方式流程图

## ■ 优点:

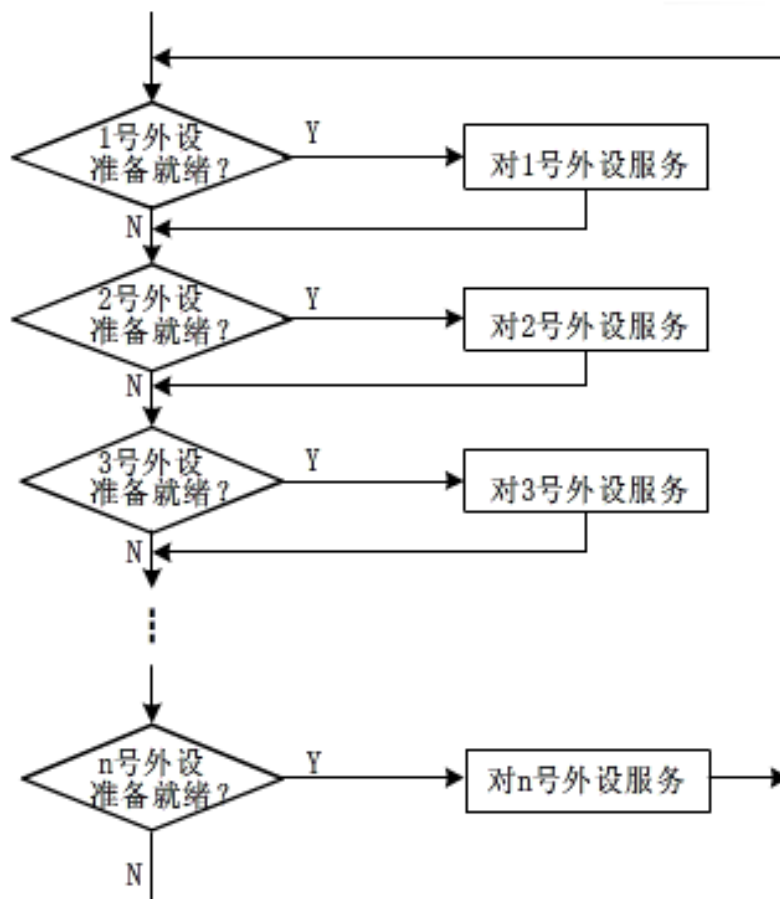
- 软硬件比较简单

## ■ 缺点:

- CPU效率低, 数据传送的实时性差, 速度较慢

## ■ 使用条件

- (1) 连接到系统的外设是简单的、慢速的, 且对实时性要求不高
- (2) 连接到同一系统的外设, 其工作速度是相近的



## ■ 举例：

- 将48000H地址中的顺序100个单元的数据发送到外设中，利用查询的方式，数据输出和查询状态共用一个地址00FFH。
- 程序如下：

```
START: MOV AX, 4000H
        MOV DS, AX
        MOV SI, 8000H
        MOV CX, 100
GOON:   MOV DX, 00FFH
WAIT:   IN AL, DX
        AND AL, 01H
```

```
        JZ WAIT
        MOV AL, [SI]
        OUT DX, AL
        INC SI
        LOOP GOON
        RET
```

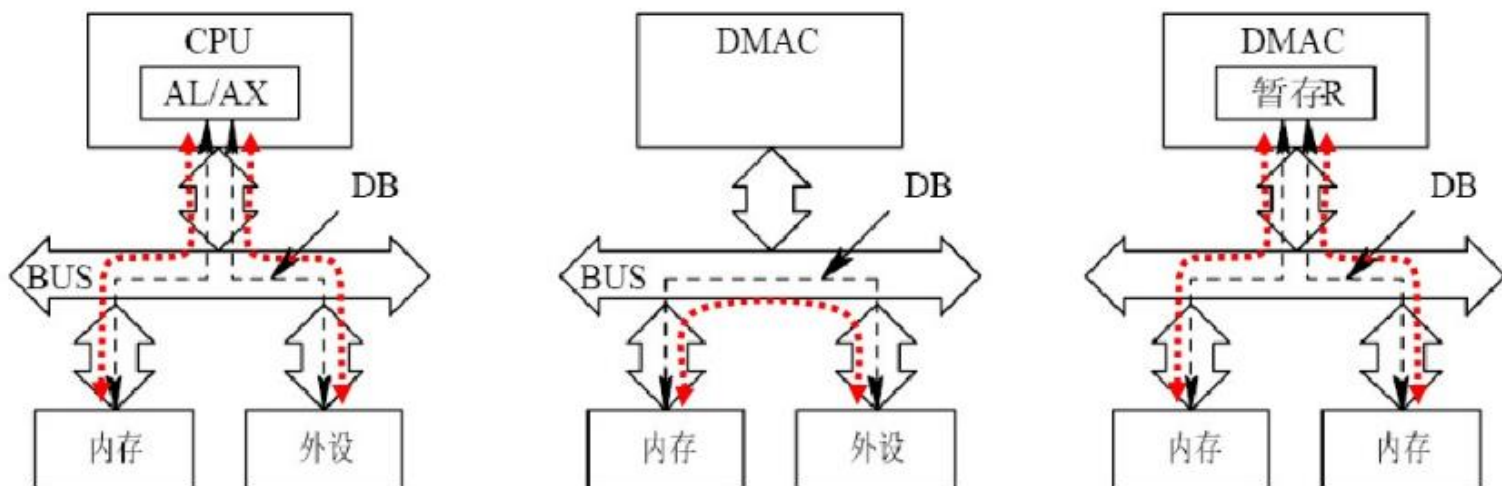
## 6.3.3 中断方式

- 程序查询传送方式明显的缺点是CPU利用率低，不能满足实时输入输出的需要，中断方式可以较好地解决这一问题。
- 所谓中断是指程序运行中出现了某种紧急事件，CPU必须中止现正在执行的程序，转去处理该紧急事件（执行一段中断服务程序），并在处理完后返回原运行的程序的过程。完整的中断处理过程包括中断请求、中断判优、中断响应、中断处理和中断返回
- 特点：
  - 外设需要在需要时向CPU提出请求，CPU再去为它服务。服务结束后或在外设不需要时，CPU回到原来被中断的地方继续执行自己的程序。
- 优点：
  - CPU效率高，实时性好
- 缺点
  - 程序编制相对较为复杂



## 6.3.4 直接存储器存取方式

- 又称为DMA方式，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。

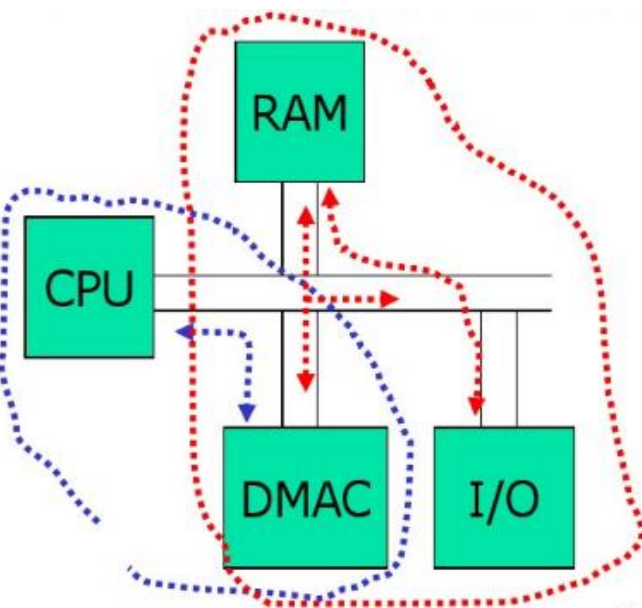


## ■ 1. DMA控制器的功能

- (1) 收到接口发出的DMA请求后，DMA控制器要向CPU发出总线请求信号HOLD，请求CPU放弃总线的控制
- (2) 当CPU响应请求并发出响应信号HLDA后，这时DMA控制器要接管总线的控制权，实现对总线的控制
- (3) 能向地址总线发出内存地址信息，找到相应单元并能够自动修改其地址计数器。
- (4) 能向存储器或外设发出读/写命令。
- (5) 能决定传送的字节数，并判断DMA传送是否结束。
- (6) 在DMA过程结束后，能向CPU发出DMA结束信号，将总线控制权交还给CPU。

## ■ 2. DMA控制器的工作过程

- (1) 当外设准备好“DMA传送请求”
- (2) DMA控制器向CPU发出“DMA请求”信号HOLD
- (3) CPU在完成当前总线操作后，对HOLD信号进行响应
- (4) DMA控制器向CPU发出响应信号DACK。

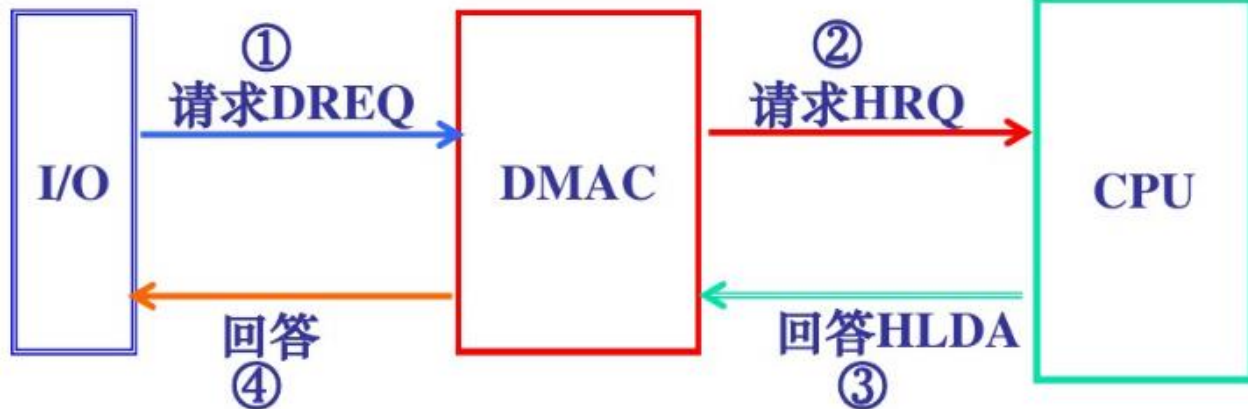


向DMA控制器发出“DMA请求”信号HOLD

A信号，对HOLD信号进行响应

总线，并向外设发出DMA

- (5) 与内存或内存
- (6) 需要重复传
- (7) 测到后就
- 撤销



内存或内存

需要重复传

测到后就

撤销

## ■ DMA存储器写的总线周期时序

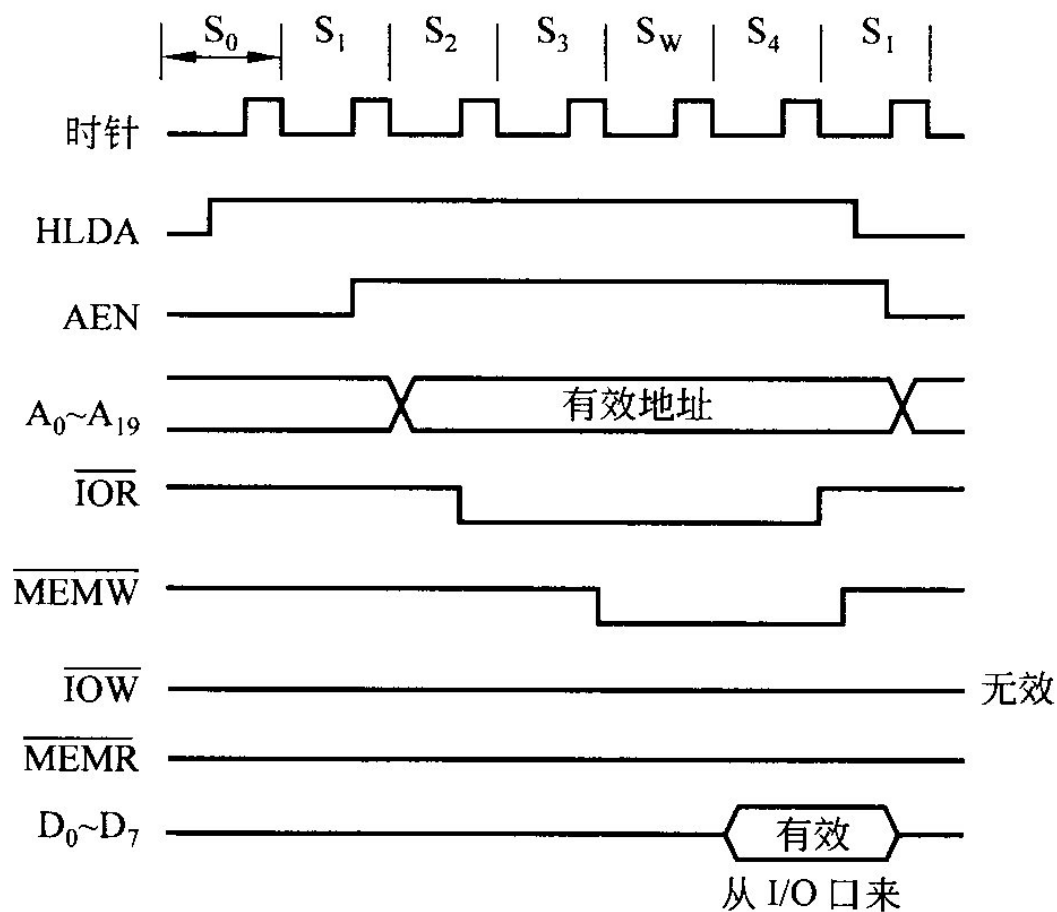


图 6-15 DMA 存储器写的总线周期时序

## ■ DMA控制方式

- 数据传输由DMA硬件来控制，数据直接在内存和外设之间交换，可以达到很高的传输速率
- 控制复杂，硬件成本相对较高。

## 6.4 中断技术

---

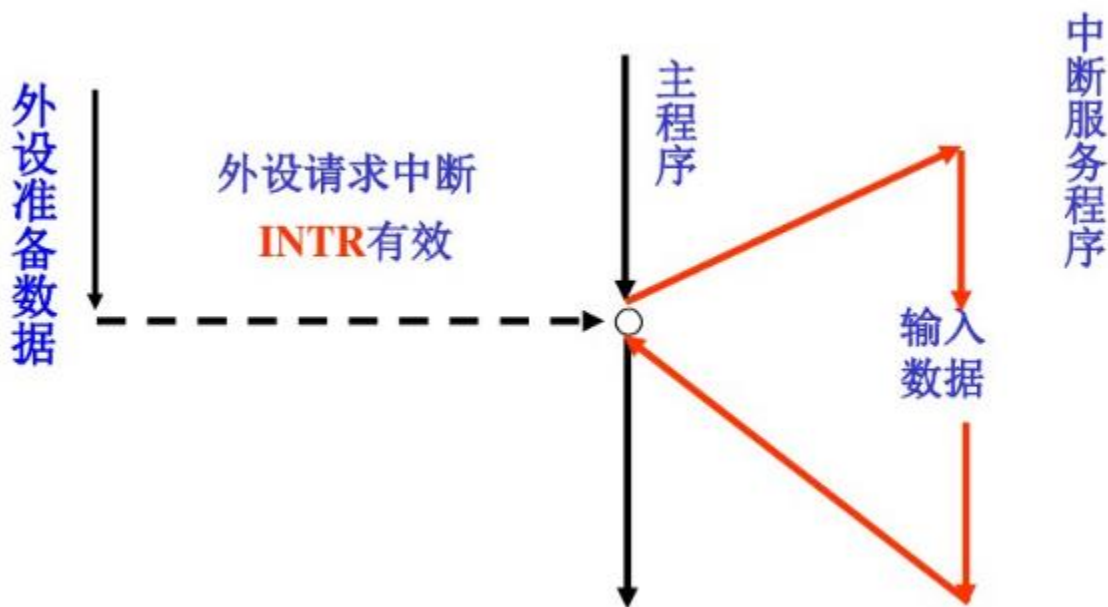
### ■ 掌握

- 中断的基本概念
- 中断响应的一般过程
- 8088/8086中断系统

## 6.4.1 中断的基本概念

### ■ 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。



## 6.4.2 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



## ■ 1. 中断请求

- INTR中断请求信号应保持到中断被处理为止
- CPU响应中断后，中断请求信号应及时撤销

## ■ 2. 中断源识别

- 软件判优：由软件来安排中断源的优先级别。顺序查询中断请求，先查询的先服务

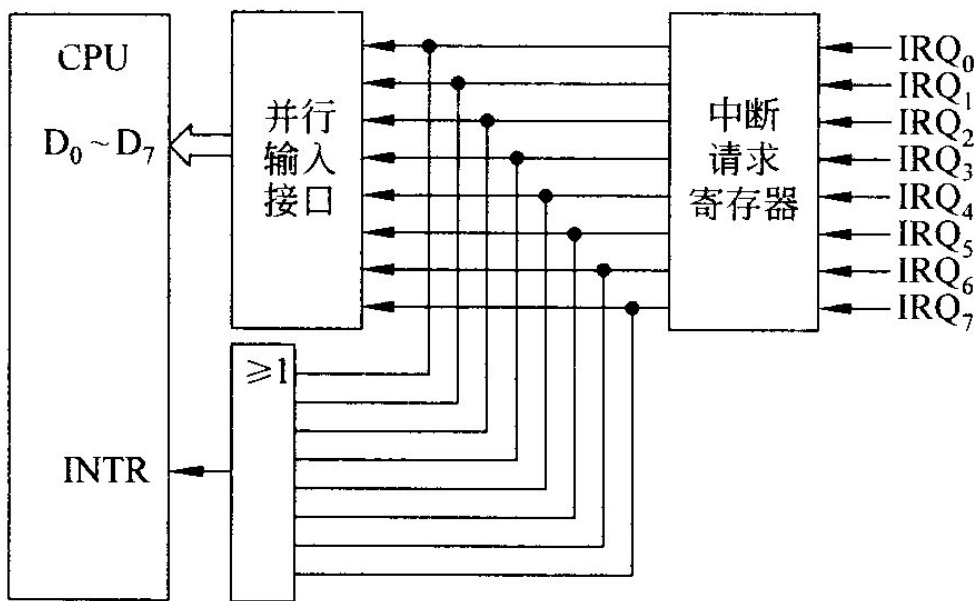


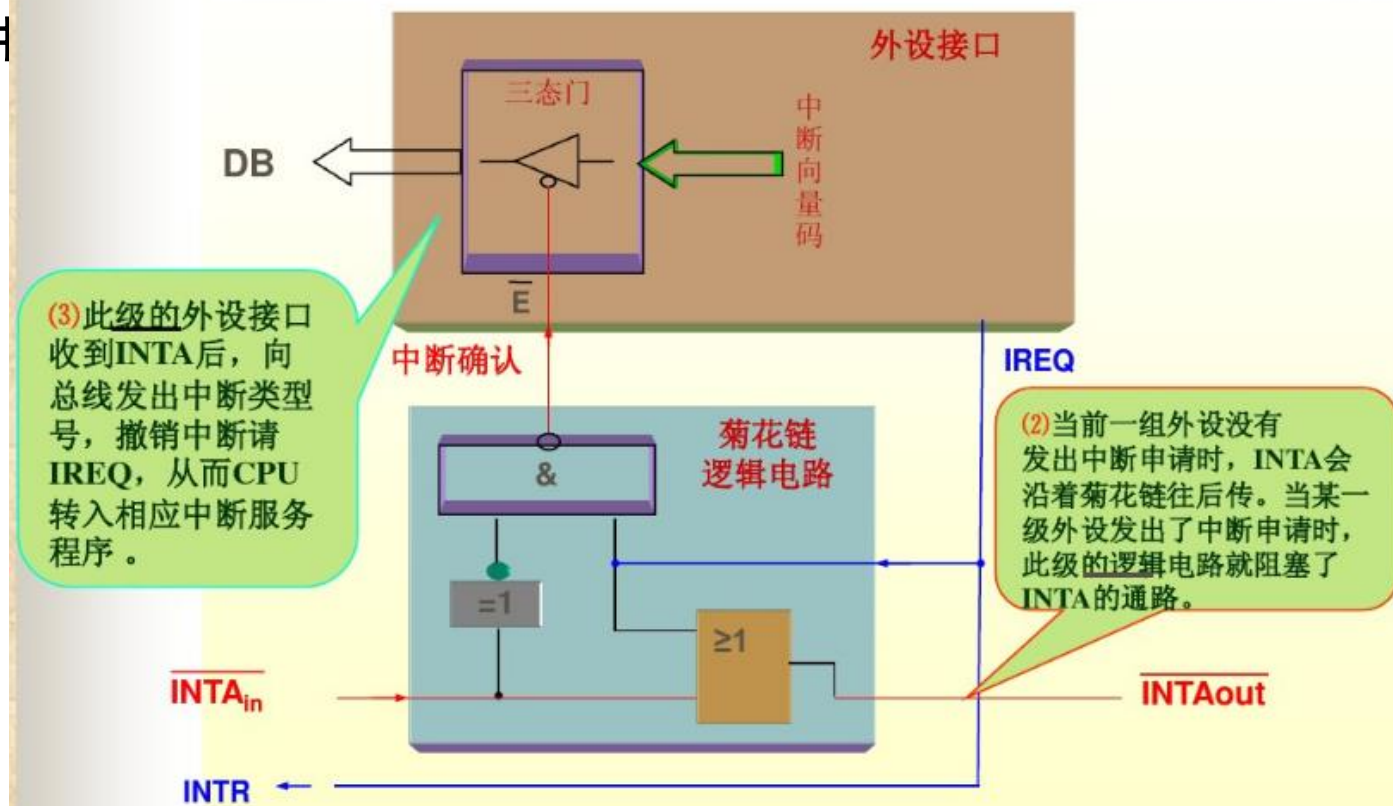
图 6-17 软件判优的电路原理图

- 硬件判优：利用专用的硬件电路或中断控制器来安排各中断源的有限界别。

#### ◆ 中断控制器判优

#### 菊花链逻辑电路

- 中



### ■ 3. 中断响应

- 向中断源发出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 ?

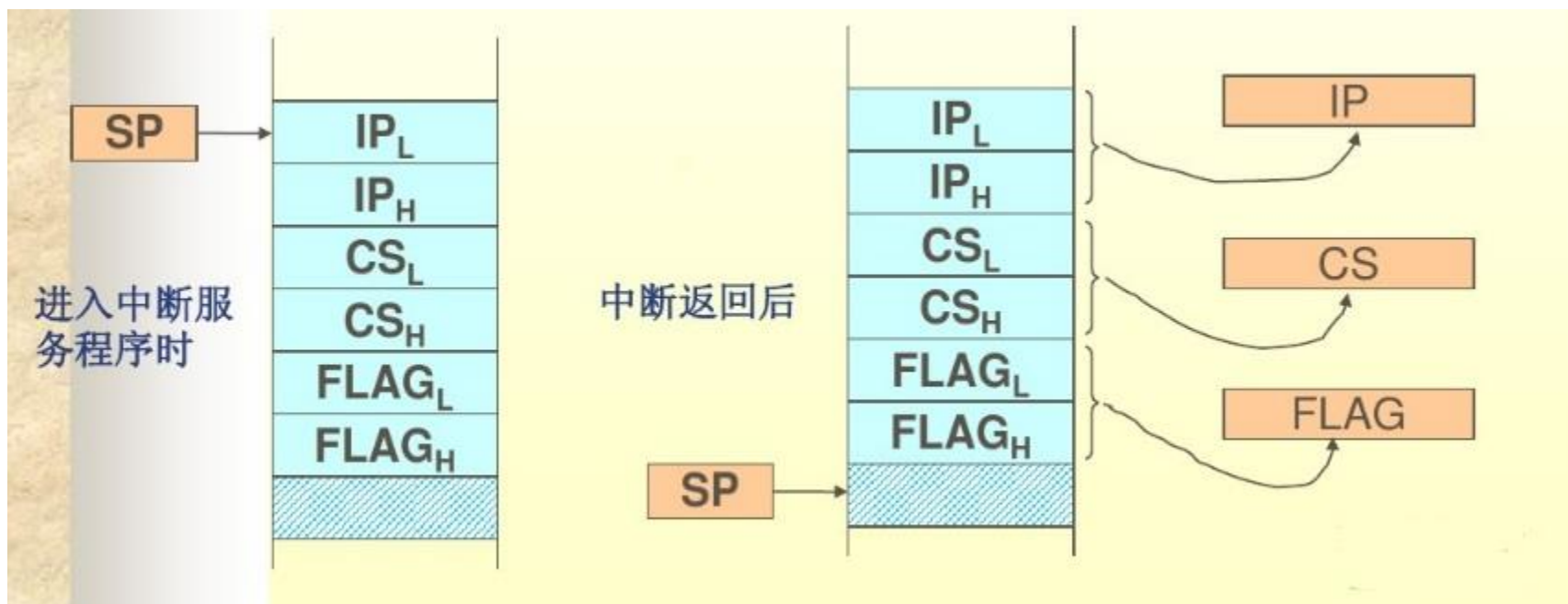
## ■ 4. 中断处理

- 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

回复断点和硬件现场



## ■ 中断处理的一般过程

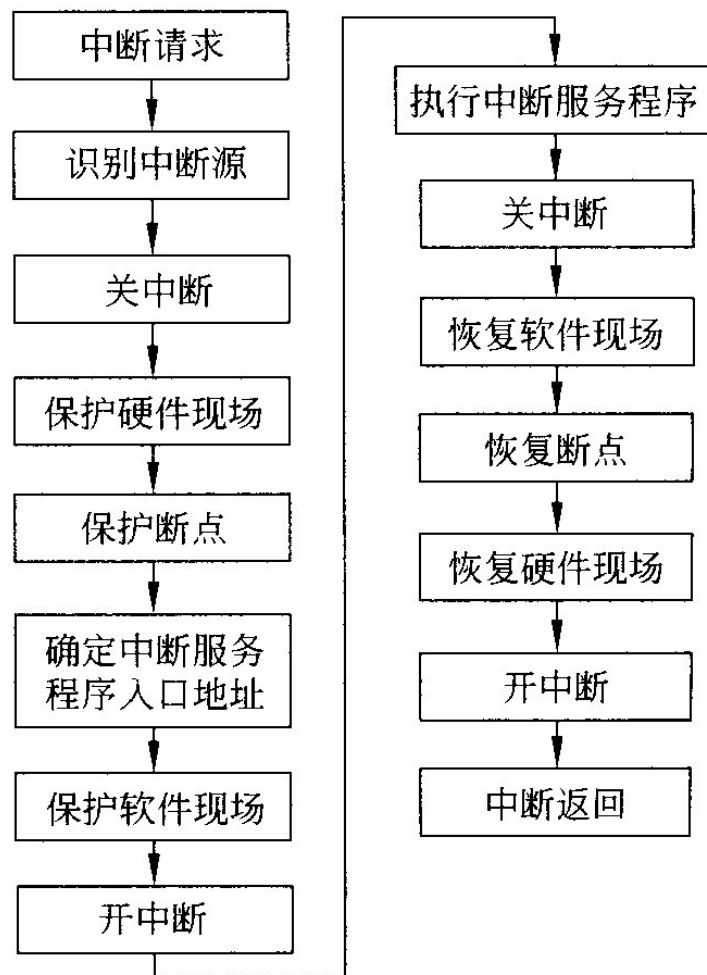
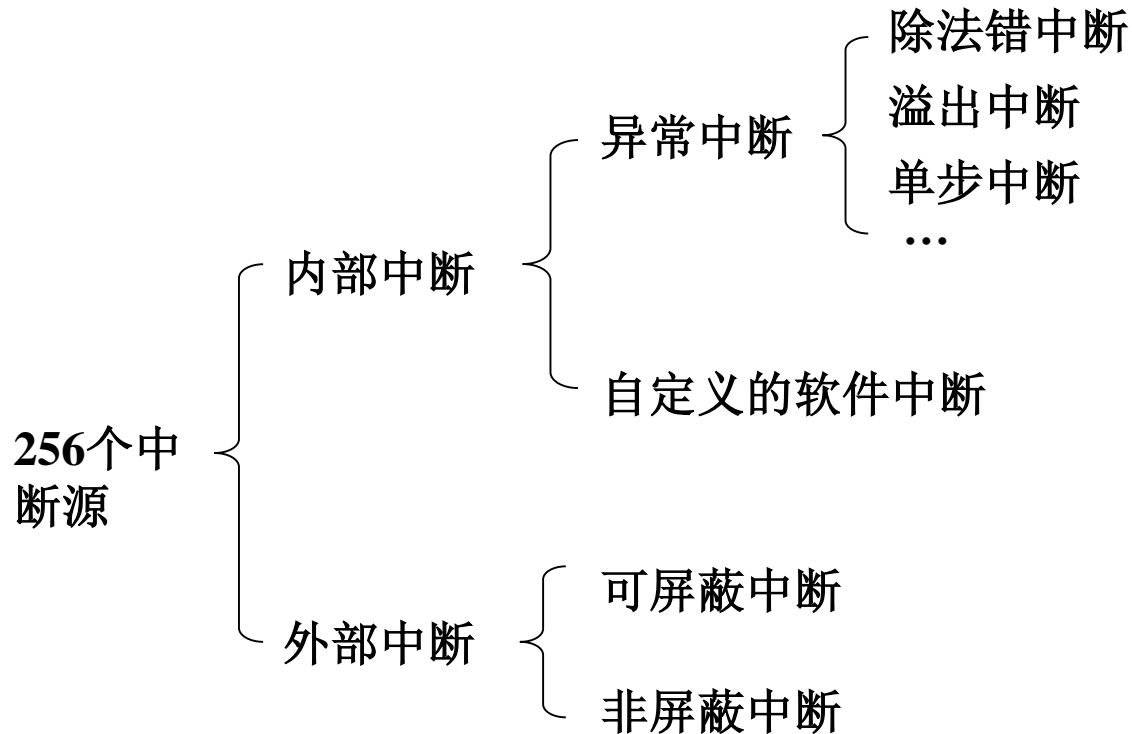


图 6-19 中断处理过程流程图

## 6.4.3 8088/8086中断系统

- 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，在算术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，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。

### ■ 3. 中断向量表

- 向量类型码：每个中断源都有一个与之对应的中断类型码。长度为1个字节。CPU可以通过中断类型码判断是哪个中断源提出的中断请求。
- 中断向量：实际上就是中断服务程序的入口地址。每个中断类型码对应一个中断向量。中断向量占4个字节存储单元，其中低位两个字节放中断向量的偏移地址（IP）；高位两个字节放中断向量的段地址（CS）。
- 中断向量表：将这些中断向量按一定的规律排列成一个表，就是所谓的中断向量表，当中断源发出中断请求时，即可查找该表，找出其中断向量，就可转入响应的中断服务子程序。
- 表的地址位于内存的00000H~003FFH，大小为1KB，共256个入口

## ■ 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/ $\overline{INTA}$ 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/ $\overline{INTA}$ 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

