



河北师范大学软件学院  
Software College of Hebei Normal University

# 计算机组成原理

## 第九章 输入输出系统

计算机的输入输出系统是整個计算机系统中~~最~~具有多样性和复杂性的部分，本章首先介绍主机与外设之间的连接问题，接着重点介绍程序查询方式、程序中断方式、DMA方式和通道方式。

# 9.1 主机与外设的连接

## 9.1.1 输入输出接口

主机和外设的连接方式有辐射型连接、总线型连接等。输入/输出接口（I/O接口）是主机和外设之间的交接界面，通过接口可以实现主机和外设之间的信息交换。

主机和外设各自具有自己的工作特点，它们在信息形式和工作速度上具有很大的差异，接口正是为了解决这些差异而设置的。

# 9.1 主机与外设的连接

主机和外设之间需要交换的信息有：

## 1. 数据信息

这类信息可以是可以通过输入设备送到计算机的输入数据，也可以是经过计算机运算处理和加工后，送到输出设备的结果数据。传送可以是并行的，也可以是串行的。

## 2. 控制信息

这是CPU对外设的控制信息或管理命令，如外设的启动和停止控制、输入或输出操作的指定、工作方式的选择、中断功能的允许和禁止等。

# 9.1 主机与外设的连接

## 3. 状态信息

这类信息用来标志外设的工作状态，比如，输入设备数据准备好标志，输出设备忙闲标志等。CPU在必要时可通过对它的查询来决定下一步的操作。

## 4. 联络信息

这是主机和外设间工作的时间配合信息，它与主机和外设间的信息交换方式密切相关。通过联络信息可以决定不同工作速度的外设和主机之间交换信息的最佳时刻，以保证整个计算机系统能统一协调地工作。

# 9.1 主机与外设的连接

## 5. 外设识别信息

这是I/O寻址的信息，使CPU能从众多的外设中寻找出与自己进行信息交换的唯一外部设备。

# 9.1 主机与外设的连接

## 9.1.2 接口的功能和基本组成

### 1. 接口的功能 ■ ■

#### (1) 实现主机和外设的通信联络控制

接口中的同步控制电路用来解决主机与外设的时间配合问题。

#### (2) 进行地址译码和设备选择

当CPU送来选择外设的地址码后，接口必须对地址进行译码以产生设备选择信息，使主机能和指定外设交换信息。

# 9.1 主机与外设的连接

## (3) 实现数据缓冲

在接口电路中，一般设置有一个或几个数据缓冲寄存器，用于数据的暂存，以避免因速度不一致而丢失数据。在传送过程中，先将数据送入数据缓冲寄存器中，然后再送到输出设备或主机中去。

## (4) 数据格式的变换

在输入或输出操作过程中，为了满足主机或外设的各自要求，接口电路中必须具有完成各类数据相互转换的功能。



# 9.1 主机与外设的连接

## (5) 传递控制命令和状态信息

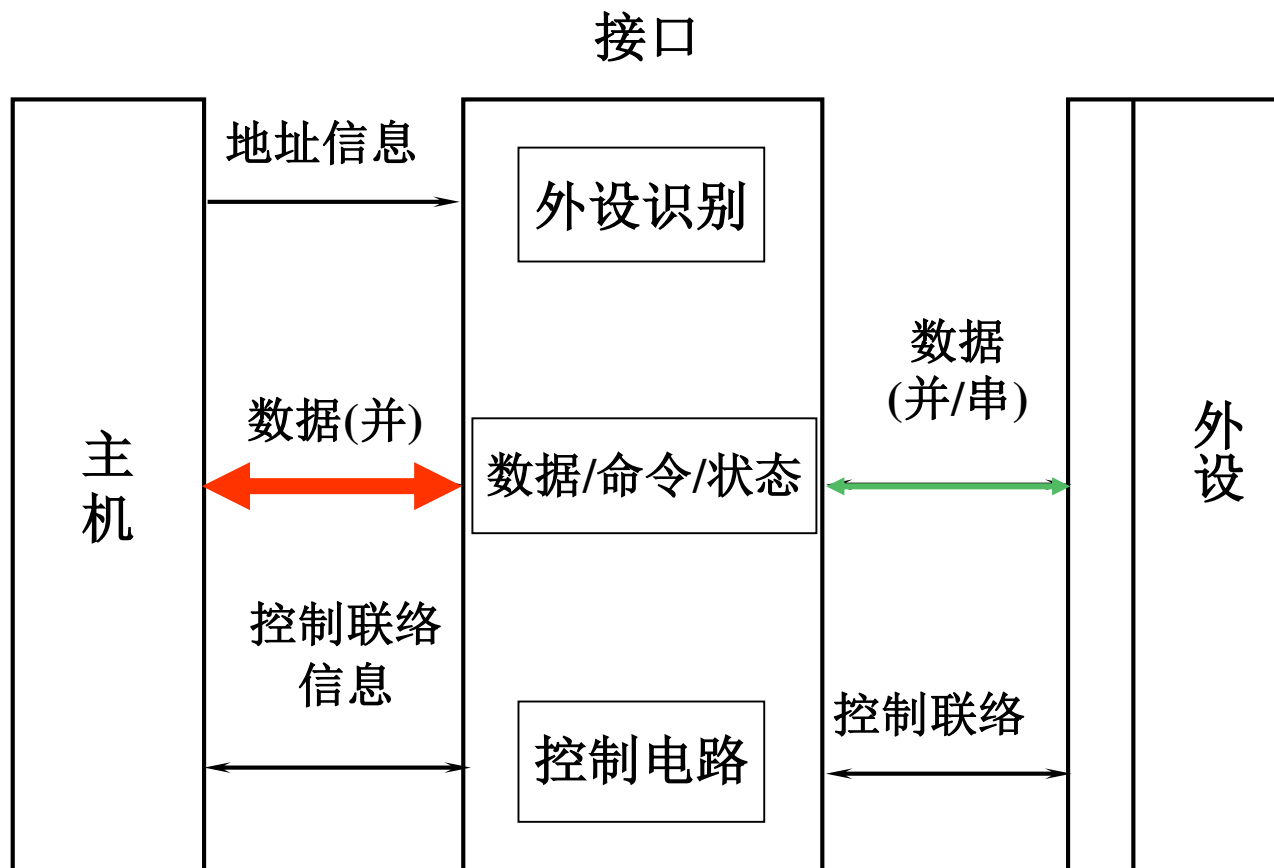
当CPU要启动某一外设时，通过接口中的控制命令寄存器向外设发出启动命令；当外设准备就绪时，则有状态信息送回接口中的状态寄存器，为CPU 提供反馈信息，告诉CPU，I/O设备已经具备和CPU交换数据的条件。当外设向CPU提出中断请求和DMA请求时，CPU也有相应的响应信号反馈给外设。

# 9.1 主机与外设的连接

## 2. 接口的基本组成 ■

接口中要分别传送数据信息、控制信息和状态信息，数据信息、控制信息和状态信息都通过数据总线来传送。大多数计算机都把I/O设备的状态信息视为输入数据，而把控制信息看成输出数据，并在接口中分设各自相应的寄存器，赋以不同的端口地址，各种信息分时地使用数据总线传送到各自的寄存器中。

# 9.1 主机与外设的连接



## 9.1 主机与外设的连接

接口与端口是两个不同的概念。端口是指接口电路中可以进行的读/写的寄存器，若干个端口加上相应的控制逻辑电路才组成接口。

## 9.1 主机与外设的连接

通常，一个接口中包含有数据端口、控制端口和状态端口。存放数据信息的寄存器称为数据端口，存放控制命令的端口称为命令端口，存放状态信息的寄存器称为状态端口。CPU通过输入指令可以从有关端口中读出信息，通过输出指令可以把信息写入有关端口。**对状态端口只进行输入操作**，将设备状态标志送到CPU中去；**对命令端口只进行输出操作**，CPU将向外设发送各种控制命令。因此，在有的接口电路中状态信息和控制信息共用一个寄存器，称之为设备的控制状态寄存器。

# 9.1 主机与外设的连接

## 3. 接口的类型

### (1) 按数据传送方式分类

有串行接口和并行接口。这里所说的数据传送方式指的是外设和接口一侧的传送方式，而在主机和接口一侧，数据总是并行传送的。

### (2) 按主机访问I/O设备的控制方式分类

可分为程序查询式接口、中断接口、DMA接口等。

### (3) 按功能选择的灵活性分类

有可编程接口和不可编程接口。

# 9.1 主机与外设的连接

## (4) 按通用性分类

有通用接口和专用接口。

## (5) 按输入/输出的信号分类

有数字接口和模拟接口。

## (6).按应用来分类

① 运行辅助接口。

② 用户交互接口。

③ 传感接口。

④ 控制接口。

# 9.1 主机与外设的连接

## 9.1.3 外设的识别与端口寻址

外设识别是通过地址总线和接口电路中的外设识别电路来实现的，**I/O**端口地址就是主机与外设直接通信的地址，**CPU**可以通过端口发送命令、读取状态和传送数据。

### 1. 端口地址编址方式

**I/O**端口编址方式有两种：一种是**I/O**映射方式，即把**I/O**端口地址与主存单元地址分别进行独立的编址；另一种是存储器映射方式，即把端口地址与主存单元地址统一编址。



# 9.1 主机与外设的连接

## (1) 独立编址 ■

主存地址空间和I/O端口地址空间是相对独立的，分别单独编址。比如，在8086中，其主存地址范围是从00000H~FFFFFFH连续的1MB，其I/O端口的地址范围从0000H~FFFFH，它们互相独立，互不影响。CPU访问主存时，由主存读/写控制线控制；访问外设时，由I/O读/写控制线控制，所以在指令系统中必须**设置专门的I/O指令**。当CPU使用I/O指令时，其指令的地址字段直接或间接的指示出端口地址。

# 9.1 主机与外设的连接

## (2) 统一编址

I/O端口地址和主存单元的地址是统一编址的，把I/O接口中的端口作为主存单元一样进行访问，**不设置专门的I/O指令。**

每个外设至少有两个寄存器：控制状态寄存器和数据缓冲寄存器，外设寄存器的地址码是连续的。在PDP-11中，把主存的高4KB地址空间留给外设接口寄存器和CPU内部寄存器使用，这4KB存储空间不允许用户再存放其他内容。

# 9.1 主机与外设的连接

## 2.独立编址方式的端口访问

Intel 80x86最多可直接寻址256个**字节端口**，  
可间接寻址65536 个**字节端口**。

任意两个连续的8位端口可作为16位端口处理；  
四个连续的8位端口可作为32位端口处理。因此，  
I/O地址空间最多能提供64K个8位端口、32K个16  
位端口、16K个32位端口或总容量不超过64KB的  
不同端口的组合。

## 9.1 主机与外设的连接

80x86的专用I/O指令IN和OUT有直接寻址和间接寻址两种类型。直接寻址I/O端口的寻址范围为00~FFH，至多为256个端口地址。这时程序可以指定：

编号0到255的256个8位端口；

编号0、2、4 ... 252、254的128个16位端口；

编号0、4、8 ... 248、252的64个32位端口。

## 9.1 主机与外设的连接

间接寻址由DX寄存器间接给出I/O端口地址。  
DX寄存器长16位，寻址范围为0000~FFFFH，最多可寻址 $2^{16}=64\text{K}$ 个端口地址，这时程序可指定：

编号0到65535的65536个8位端口；

编号0、2、4 ... 65532、65534的32768个16位  
端口；

编号0、4、8 ... 65528、65532的16384个32位  
端口。

## 9.1 主机与外设的连接

**CPU一次可实现字节（8位）、字（16位）或双字（32位）的数据传送，与存储器中的双字一样。32位端口应对准可被4整除的偶地址，与存储器中的字一样，16位端口应对准偶地址，8位端口可定位在偶地址，也可定位在奇地址。**

# 9.1 主机与外设的连接

## 9.1.4 输入/输出信息传送控制方式

主机和外设之间的信息传送控制方式，经历了由低级到高级、由简单到复杂、由集中管理到各部件分散管理的发展过程，按其发展的先后次序和主机与外设并行工作的程度，可以分为四种。

# 9.1 主机与外设的连接

## 1. 程序查询方式

程序查询方式是一种程序直接控制方式，这是主机与外设间进行信息交换的最简单方式，输入和输出完全是通过CPU执行程序来完成的。

这种方式控制简单，但外设和主机不能同时工作，各外设之间也不能同时工作，系统效率很低，因此，仅适用于外设的数目不多，对I/O处理的实时要求不那么高，CPU的操作任务比较单一，并不很忙的情况。



# 9.1 主机与外设的连接

## 2. 程序中断方式 ■

外在作好输入/输出准备时，向主机发中断请求，主机接到请求后就暂时中止原来执行的程序，转去执行中断服务程序对外部请求进行处理，在中断处理完毕后返回原来的程序继续执行。

程序中断不仅允许主机和外设同时并行工作，并且允许一台主机管理多台外设。但是完成一次程序中断需要许多辅助操作，可能使CPU应接不暇；对于一些高速外设，可能会造成信息丢失，因此，它主要适用于中、低速外设。

# 9.1 主机与外设的连接

## 3. 直接存储器存取（DMA）方式

**DMA方式是在主存储器和外部设备之间开辟直接的数据通路，可以进行基本上不需要CPU介入的主存和外设之间的信息传送，这样不仅能保证CPU的高效率，而且能满足高速外设的需要。**

**DMA方式只能进行简单的数据传送操作，在数据块传送的起始和结束时还需CPU及中断系统进行预处理和后处理。**

# 9.1 主机与外设的连接

## 4. I/O通道控制方式 ■

通道是一个具有特殊功能的处理器，它能独立地执行通道程序，产生相应的控制信号，实现对外设的统一管理和外设与主存之间的数据传送。但它不是一个完全独立的处理机，它要在CPU的I/O指令指挥下才能启动、停止或改变工作状态，是从属于CPU的一个专用处理器。

一个通道执行输入/输出过程全部由通道按照通道程序自行处理，不论交换信息多少，只打扰CPU两次（启动和停止时）。

## 9.2 程序查询方式及其接口

### 9.2.1 程序查询方式

#### 1. 程序查询的基本思想 ■

由CPU执行一段输入、输出程序来实现主存与外设之间的数据传送方式，叫做程序直接控制方式。根据外设的不同性质，这种传送方式又可分为无条件传送和程序查询方式两种。

在无条件传送方式中，I/O接口总是准备好接收主机的输出数据，或总是准备好向主机输入的数据，因而CPU无需查询外设的工作状态，而默认外设始终处于准备就绪状态。

## 9.2 程序查询方式及其接口

许多外设的工作状态是很难事先预知的，为了保证数据传送的正确进行，就要求CPU在程序中查询外设的工作状态，如果外设尚未准备就绪，CPU就等待，只有外设已作好准备，CPU才能执行I/O指令，这就是程序查询方式。

## 9.2 程序查询方式及其接口

### 2. 程序查询方式的工作流程

#### (1) 预置传送参数

在传送数据之前，由CPU执行一段程序，预置传送参数。传送参数包括存取数据的主存缓冲区首地址和传送数据的个数。

#### (2) 向I/O接口发命令字

当CPU选中某台外设时，执行输出指令向I/O接口发出命令字，启动外设，为接收数据或发送数据的操作做准备。

## 9.2 程序查询方式及其接口

### (3) 从I/O接口取回状态字

CPU执行输入指令，从I/O接口中取回状态字并进行测试，判断数据传送是否可以进行。

### (4) 查询外设标志

CPU不断查询状态标志，如果外设没有准备就绪，CPU就踏步进行等待，一直到这个外设准备就绪，并发出“准备就绪”信号为止。

## 9.2 程序查询方式及其接口

### (5) 传送数据

只有外设准备好，才能实现主机与外设间的一次数据传送。输入时，CPU执行输入指令，从I/O接口的数据缓冲寄存器中接收数据；输出时，CPU执行输出指令，将数据写入I/O接口的数据缓冲寄存器。

### (6) 修改传送参数

每进行一次数据传送，需要修改传送参数，其中包括主存缓冲区地址加1，传送个数减1。

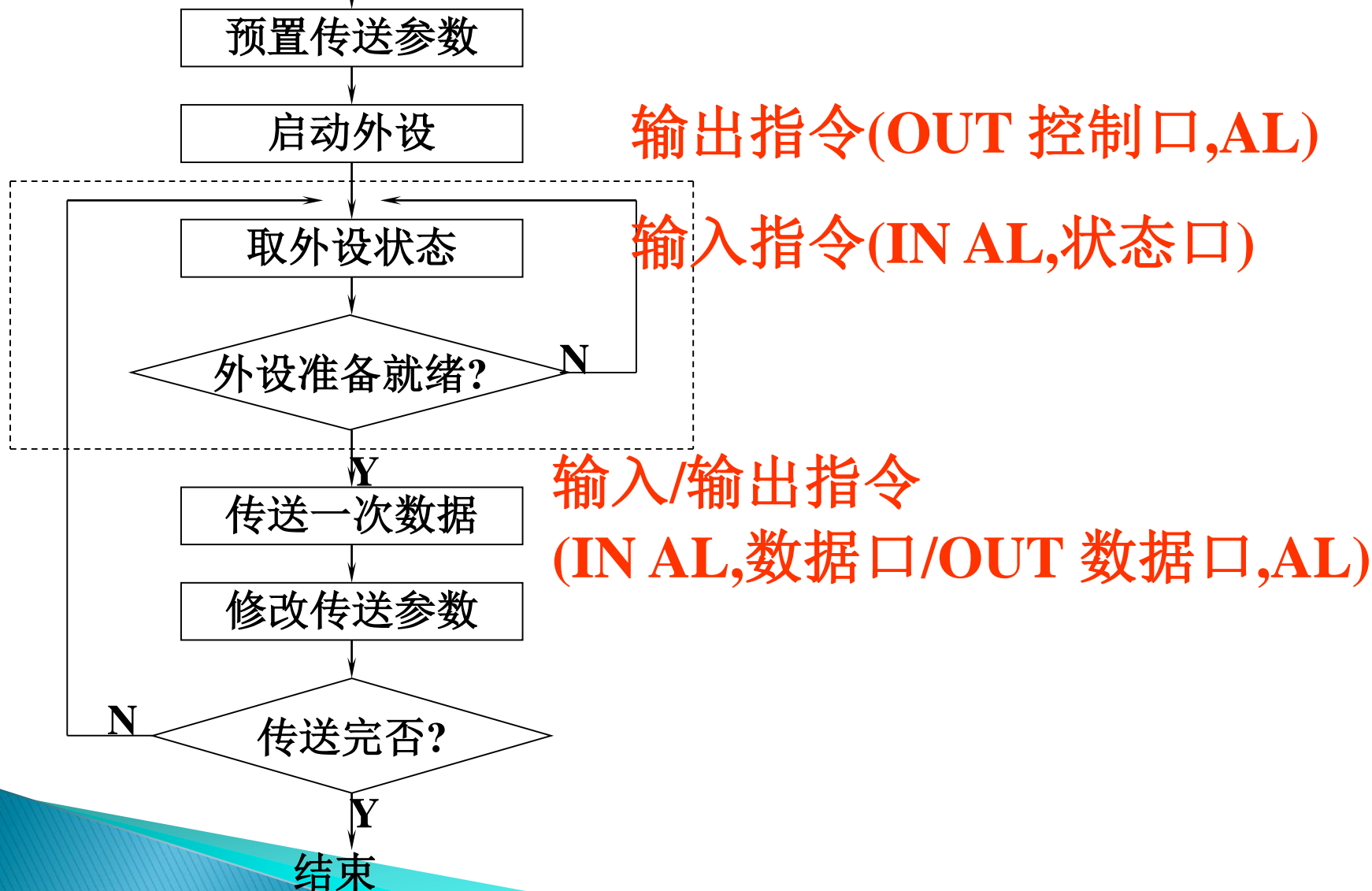


## 9.2 程序查询方式及其接口

### (7) 判断传送是否结束

如果传送个数不为0，则转第3步，继续传送，直到传送结束为止。

## 9.2 程序查询方式及其接口



## 9.2 程序查询方式及其接口

### 9.2.2 程序查询方式接口

最简单、经济的I/O方式，只需很少的硬件。

通常接口中至少有两个寄存器，一个是数据缓冲寄存器，即数据端口，用来存放与CPU进行传送的数据信息，另一个是供CPU查询的设备状态寄存器，即状态端口，这个寄存器由多个标志位组成，其中最重要的是设备准备就绪标志。当CPU得到这位信息后就进行判断，以决定下一步是继续循环等待还是进行I/O传送，也有些计算机仅设置状态标志触发器，其作用与设备状态寄存器相同。

## 9.3 中断系统和程序中中断方式

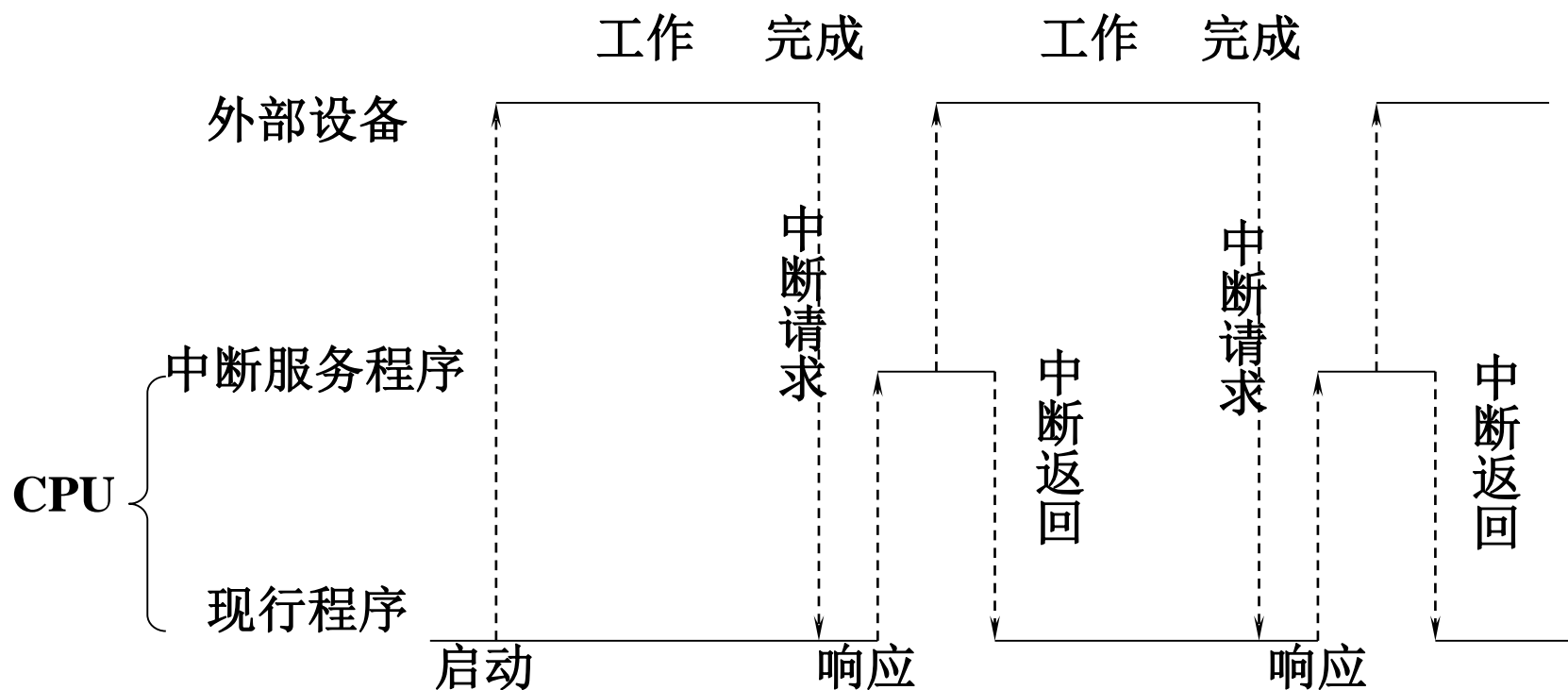
### 9.3.1 中断的基本概念

#### 1. 中断的提出 ■ ■

程序查询方式存在着下列明显的缺点。

- ① 在查询过程中，**CPU**长期处于踏步等待状态，使系统效率大大降低。
- ② **CPU**在一段时间内只能和一台外设交换信息，其它设备不能同时工作。
- ③ 不能发现和处理预先无法估计的错误和异常情况。

## 9.3 中断系统和程序中中断方式



## 9.3 中断系统和程序中中断方式

为了提高输入/输出能力和CPU的效率，50年代中期，中断传送方式被引进计算机系统。

现代计算机，无论是巨型机、大型机、小型机还是微型机无不具有中断能力。

中断系统是计算机实现中断功能的软、硬件总称。一般在CPU中配置中断机构，在外设接口中配置中断控制器，在软件上设计相应的中断服务程序。

## 9.3 中断系统和程序中中断方式

### 2. 程序中断与调用子程序的区别 ■

表面上看起来，计算机的中断处理过程有点类似于调用子程序的过程，这里现行程序相当于主程序，中断服务程序相当于子程序。但是，它们之间却是有着本质上的区别的。

## 9.3 中断系统和程序中中断方式

- (1) 子程序的执行是由程序员事先安排好的（由一条调用子程序指令转入），而中断服务程序的执行则是由随机的中断事件引起的；
- (2) 子程序的执行受到主程序或上层子程序的控制，而中断服务程序一般与被中断的现行程序毫无关系；
- (3) 不存在同时调用多个子程序的情况，而有可能发生多个外设同时请求CPU为自己服务的情况。



## 9.3 中断系统和程序中中断方式

### 3. 中断的基本类型

#### (1) 自愿中断和强迫中断

自愿中断又称程序自中断，它不是随机产生的中断，而是在程序中安排的有关指令，这些指令可以使机器进入中断处理的过程，如：指令系统中的软件中断指令等。

**强迫中断**是随机产生的中断，不是程序中事先安排好的。当这种中断产生后，由中断系统强迫计算机中止现行程序并转入中断服务程序。

## 9.3 中断系统和程序中中断方式

### (2) 程序中中断和简单中断

**程序中中断**就是我们前面提到的中断，主机在响应中断请求后，通过执行一段中断服务程序来处理更紧迫的任务。

简单中断就是外设与主存间直接进行信息交换的方法，即**DMA**方式。这种“中断”不去执行中断服务程序，故不破坏现行程序的状态。主机发现有简单中断请求（也就是**DMA**请求）时，将让出一个或几个存取周期供外设与主存交换信息，然后继续执行程序。

## 9.3 中断系统和程序中中断方式

### (3) 内中断和外中断

内中断是指由于CPU内部硬件或软件原因引起的中断。

**外中断**是指CPU以外的部件引起的中断。

### (4) 向量中断和非向量中断

**向量中断**是指那些中断服务程序的入口地址是由中断事件自己提供的中断。中断事件在提出中断请求的同时，通过硬件向主机提供中断服务程序入口地址，即向量地址。

## 9.3 中断系统和程序中中断方式

**非向量中断**的中断事件不能直接提供中断服务程序的入口地址，而由CPU 查询之后得到。

### (5) 单重中断和多重中断

**单重中断**在CPU执行中断服务程序的过程中不能被再打断。

**多重中断**在执行某个中断服务程序的过程中，CPU 可去响应级别更高的中断请求，又称为中断嵌套。

## 9.3 中断系统和程序中中断方式

### 9.3.2 中断请求和中断判优

#### 1. 中断源和中断请求信号

**中断源是指中断的来源，即任何引起计算机中断的事件**，一般计算机都有多个中断源。由于每个中断源向CPU发出中断请求的时间是随机的，为了记录中断事件并区分不同的中断源，可采用具有存储功能的触发器来记录中断源，称为中断请求触发器。当某一个中断源有中断请求时，其相应的中断请求触发器置成“1”状态，此时，该中断源向CPU发出中断请求信号。

## 9.3 中断系统和程序中中断方式

多个中断请求触发器构成一个中断请求寄存器，其中每一位对应一个中断源，中断请求寄存器的内容称为中断字或中断码，中断字中为“1”的位就表示对应的中断源有中断请求。

## 9.3 中断系统和程序中中断方式

### 2. 中断请求信号的传送 ■

#### (1) 独立请求线

每个中断源单独设置中断请求线，将中断请求信号直接送往CPU，这种方式的特点是CPU在接到中断请求的同时也就知道了中断源是谁，其中断服务程序的入口地址在哪里。

## 9.3 中断系统和程序中中断方式

### (2) 公共请求线

多个中断源共有一根公共请求线，这种方式的特点是在负载允许的情况下，中断源的数目可随意扩充，但CPU在接到中断请求后，必须通过软件或硬件的方法来识别中断源，然后再找出中断服务程序的入口地址。

### (3) 二维结构

将中断请求线连成二维结构，同一优先级别的中断源，采用一根公共的请求线，不同请求线上的中断源优先级别不同，这种方式综合了前两种方式的优点，在中断源较多的系统中常采用这种方式。



## 9.3 中断系统和程序中中断方式

### 3. 中断优先级与判优方法 ■

当多个中断源同时发出中断请求时，CPU在任何瞬间只能接受一个中断源的请求。通常，把全部中断源按中断的性质和处理的轻重缓急安排优先级，并进行排队。

确定中断优先级的原则是：对那些提出中断请求后需要立刻处理，否则就会造成严重后果的中断源规定最高的优先级；而对那些可以延迟响应和处理的断源规定较低的优先级。如故障中断一般优先级较高，接着才是I/O设备中断。而在I/O设备中又可以根据各个设备的速度来决定优先级。

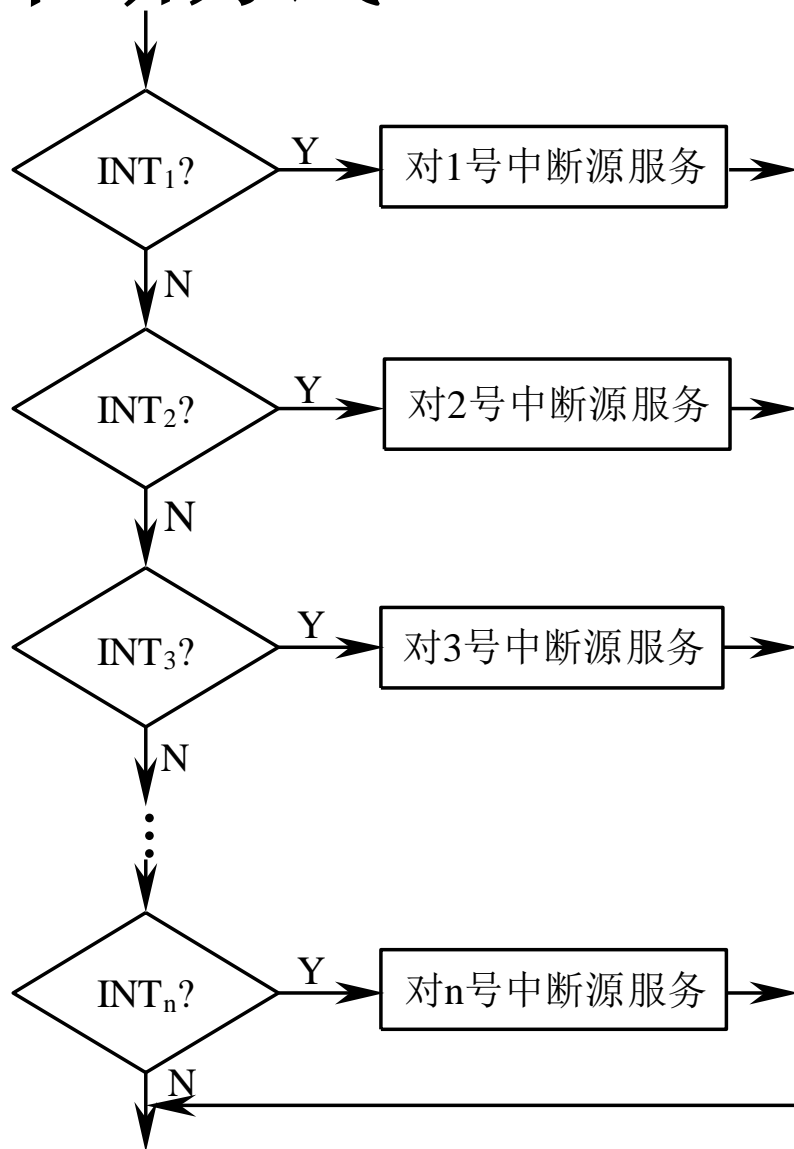
## 9.3 中断系统和程序中中断方式

每个中断源均有一个为其服务的中断服务程序，每个中断服务程序都有与之对应的优先级别。另外，CPU正在执行的程序也有优先级。只有当某个中断源的优先级别高于CPU现在的优先级时，才能中止CPU执行现在的程序。

## 9.3 中断系统和程序中中断方式

### (1) 软件判优法

软件判优法，就是用程序来判别优先级，这是最简单的中断判优方法。



## 9.3 中断系统和程序中中断方式

当CPU接到中断请求信号后，就执行查询程序，逐个检测中断请求寄存器的各位状态，检测顺序是按优先级的大小排列的，最先检测的中断源具有最高的优先级，其次检测的中断源具有次高优先级，如此下去，最后检测的中断源具有最低的优先级。

显然，软件判优是与识别中断源结合在一起的，当查询到中断请求信号的发出者，也就是找到了中断源，程序立即可以转入对应的中断服务程序中去。

## 9.3 中断系统和程序中中断方式

### (2) 硬件判优电路 ■

采用硬件实现中断优先级判定可节省CPU时间，而且速度快，但是成本较高。

根据中断请求信号的传送方式不同，有不同的优先排队电路，常见的有以下几种方案。

独立请求线的优先排队电路

公共请求线的优先排队电路

## 9.3 中断系统和程序中中断方式

### 9.3.3 中断响应和中断处理

#### 1. CPU响应中断的条件 ■

##### (1) CPU接收到中断请求信号

首先中断源要发出中断请求，同时CPU还要接收到这个中断请求信号。

##### (2) CPU允许中断

CPU允许中断即开中断。CPU内部有一个中断允许触发器，只有当其被置位时，CPU才可能响应中断源的中断请求（中断开放）。如其被复位，CPU处于不可中断状态，即使中断源有中断请求，CPU也不响应（中断关闭）。

## 9.3 中断系统和程序中中断方式

### (3) 一条指令执行完毕

一般情况下，CPU在一条指令执行完毕，且没有更紧迫的任务时才能响应中断请求。

### 2. 中断隐指令 ■

CPU响应中断之后，经过某些操作，转去执行中断服务程序。这些操作是由硬件直接实现的，我们把它称为中断隐指令。中断隐指令并不是指令系统中的一条真正的指令，它没有操作码，所以中断隐指令是一种不允许、也不可能为用户使用的特殊指令。其所完成的操作主要有：

## 9.3 中断系统和程序中中断方式

### (1) 保存断点

将原来程序的断点（即程序计数器PC的内容）保存起来。

### (2) 暂不允许中断

为了在用软件保护中断现场（即CPU 的主要寄存器状态）时，不被新的中断所打断，从而保证被中断的程序在中断服务程序执行完毕之后能接着正确地执行下去。

### (3) 引出中断服务程序

引出中断服务程序的实质就是取出中断服务程序的入口地址送程序计数器。



## 9.3 中断系统和程序中中断方式

### 3. 中断周期 ■

中断周期需完成如下操作：

- (1) 将特定地址“0”送至存储器地址寄存器，记作  $0 \rightarrow \text{MAR}$ ;
- (2) 将 PC 的内容（断点）送至 MDR，记作  $(\text{PC}) \rightarrow \text{MDR}$ ;
- (3) 向主存发写命令，启动存储器做写操作，记作 **Write**;
- (4) 将MDR的内容通过数据总线写入到MAR所指示的主存单元（0号）中，记作  $\text{MDR} \rightarrow \text{M}(\text{MAR})$ ;

## 9.3 中断系统和程序中中断方式

- (5) 向量地址形成部件的输出送至PC，为进入中断服务程序作准备，记作向量地址 $\rightarrow$ PC；
  - (6) 关中断，将中断允许触发器清0，记作 $0\rightarrow$ EINT。
- 如果断点存入堆栈，只需将上述(1)改为堆栈指针 $SP\rightarrow$ MAR。

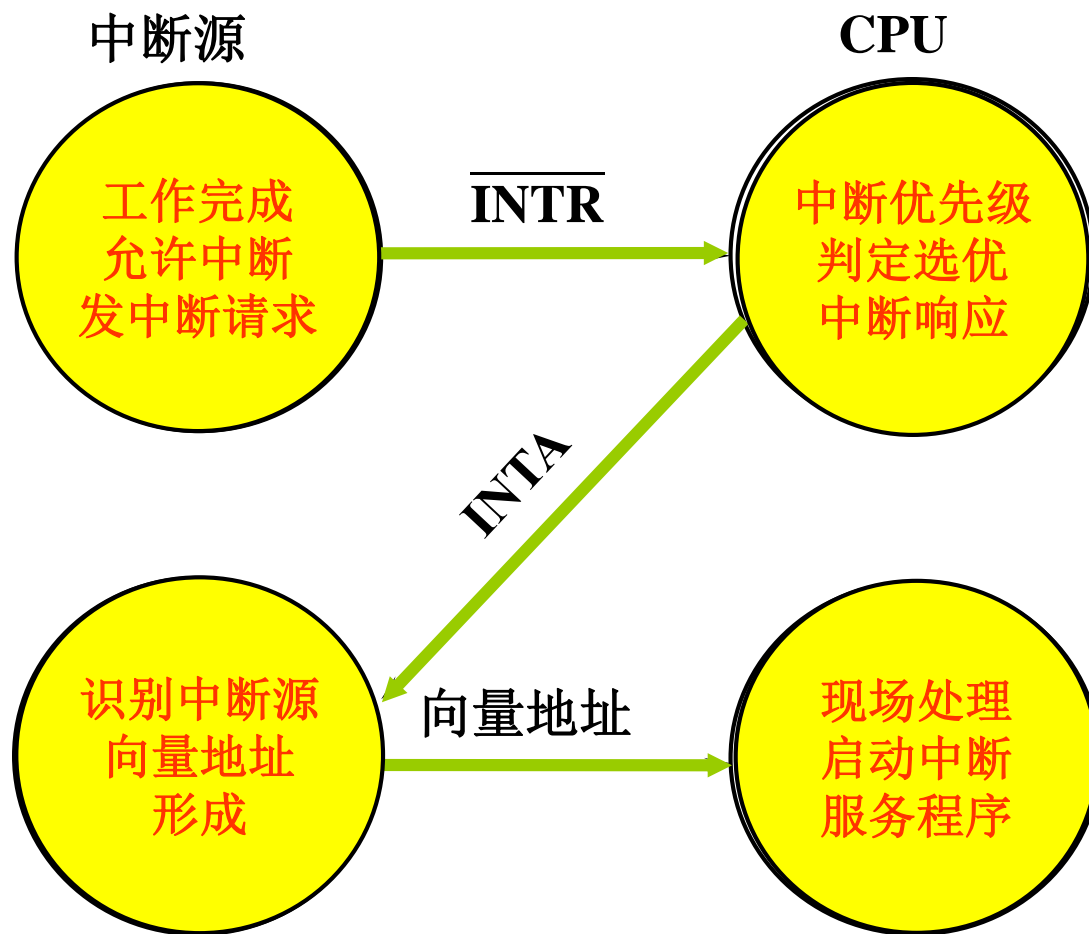
## 9.3 中断系统和程序中中断方式

### 4. 进入中断服务程序

识别中断源在于转入为该中断源专门设置的中断服务程序。

向量中断时，中断源向CPU发出中断请求信号之后，CPU经过一定的判优处理，若决定响应这个中断请求，则向中断源发出中断响应信号。中断源接到中断响应信号后就通过自己的向量地址发生器向CPU发送向量地址。

## 9.3 中断系统和程序中中断方式



## 9.3 中断系统和程序中中断方式

向量地址通常有两种情况：

### (1) 向量地址是中断服务程序的入口地址

如果向量地址就是中断服务程序的入口地址，则CPU 不需要再经过处理就可以进入相应的中断服务程序。

$PC \leftarrow 8 \times NNN$  转中断服务程序入口地址 ■

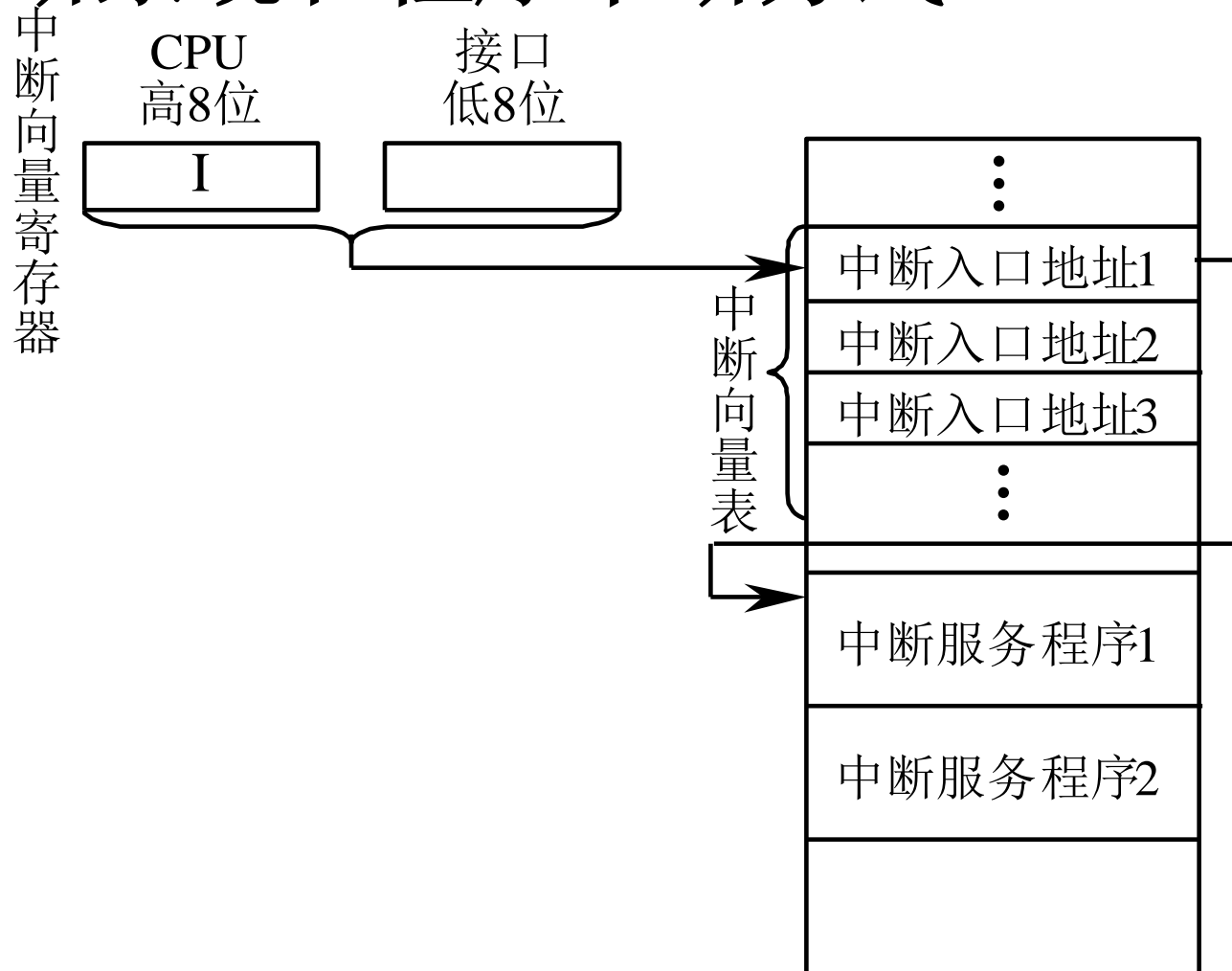
由此可见，中断服务程序的入口地址依次是00H、08H、10H、.....、38H。

## 9.3 中断系统和程序中中断方式

### (2) 向量地址是中断向量表的指针

如果向量地址是中断向量表的指针，则向量地址指向一个中断向量表，从中断向量表的相应单元中再取出中断服务程序的入口地址，此时中断源给出的向量地址是中断服务程序入口地址的地址。

## 9.3 中断系统和程序中中断方式



## 9.3 中断系统和程序中中断方式

### 5. 中断现场的保护和恢复 ■

中断现场指的是发生中断时CPU的主要状态，其中最重要的是断点，另外还有一些通用寄存器因程序运行状态转换。一般说来，在中断隐指令中，CPU硬件将自动保存断点，有些计算机还自动保存程序状态寄存器的内容。但是，在许多应用中，仅保存这一、二个寄存器的内容是不够的。

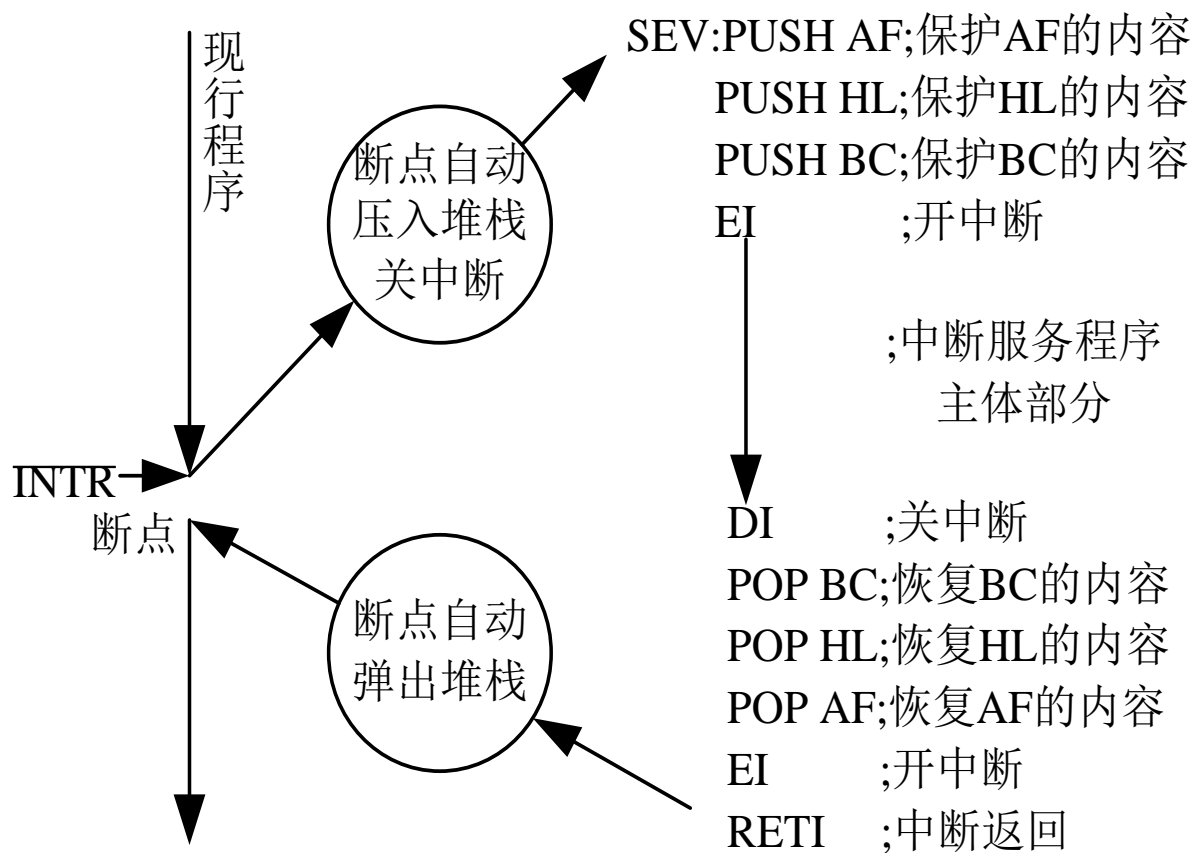


## 9.3 中断系统和程序中中断方式

为此，在中断服务程序开始时，应由软件去保存那些硬件没有保存，而在中断服务程序中又可能用到的寄存器（如某些通用寄存器）的内容，在中断返回之前，这些内容还应该被恢复。

现代计算机一般都先采用硬件方法来自动快速的保护和恢复部分重要的现场，其余寄存器的内容再由软件完成保护和恢复，这种方法的硬件支持是堆栈。

## 9.3 中断系统和程序中中断方式



## 9.3 中断系统和程序中中断方式

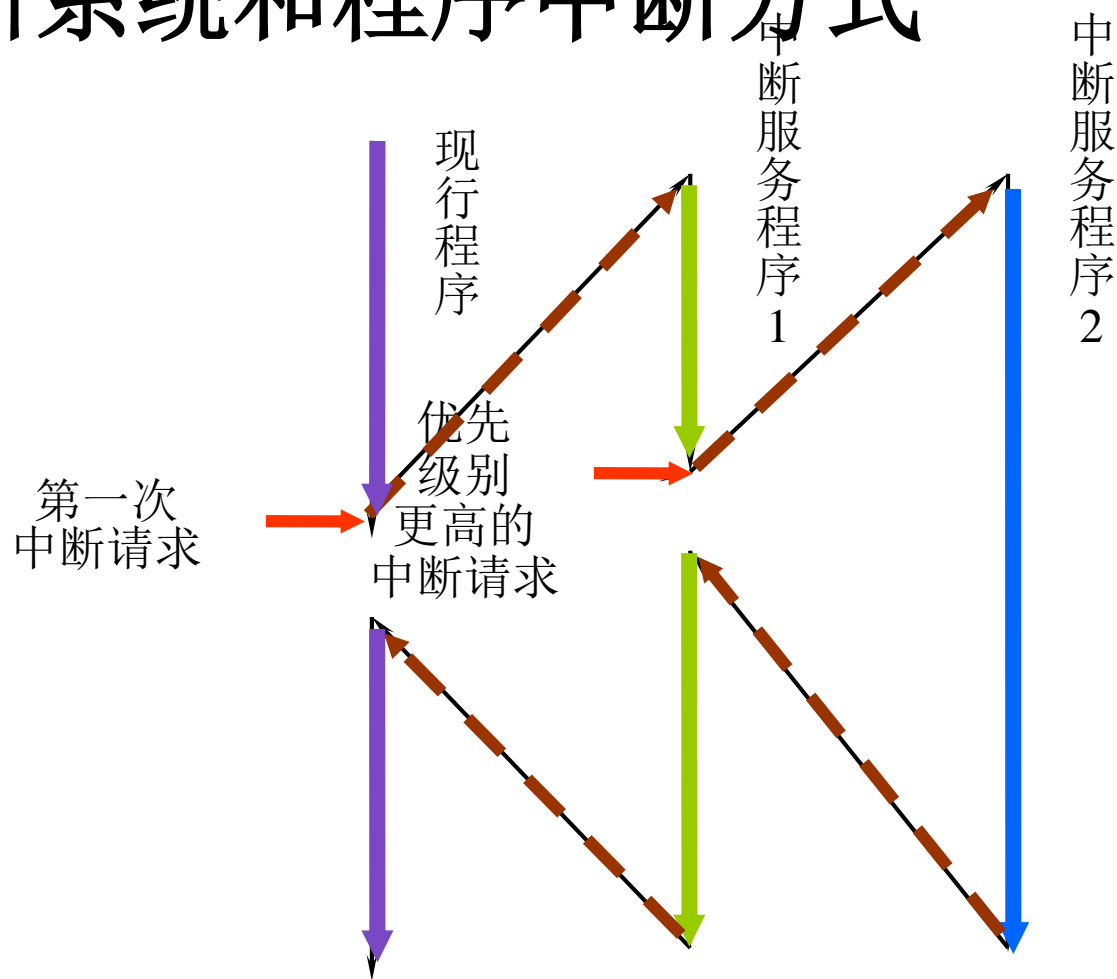
### 9.3.4 多重中断与中断屏蔽

#### 1. 中断嵌套 ■

中断嵌套的层次可以有 multiple 层，越在里层的中断越急迫，优先级越高，因此优先得到CPU的服务。

要使计算机具有多重中断的能力，首先要能保护多个断点，先发生的中断请求的断点，先保护后恢复；后发生的中断请求的断点，后保护先恢复，堆栈的先进后出特点正好满足多重中断这一先后次序的需要。在CPU进入某一中断服务程序之后，系统必须处于开中断状态，否则中断嵌套是不可能实现的。

## 9.3 中断系统和程序中中断方式



## 9.3 中断系统和程序中中断方式

### 2. 允许和禁止中断 ■

允许中断还是禁止中断是用CPU中的中断允许触发器控制的，当中断允许触发器被置“1”，则允许中断，当中断允许触发器被置“0”，则禁止中断。

允许中断即开中断，下列情况时应开中断：

- (1) 在中断服务程序执行完毕，恢复中断现场之后；
- (2) 在多重中断的情况下，保护中断现场之后。

## 9.3 中断系统和程序中断方式

禁止中断即关中断，下列情况时应关中断：

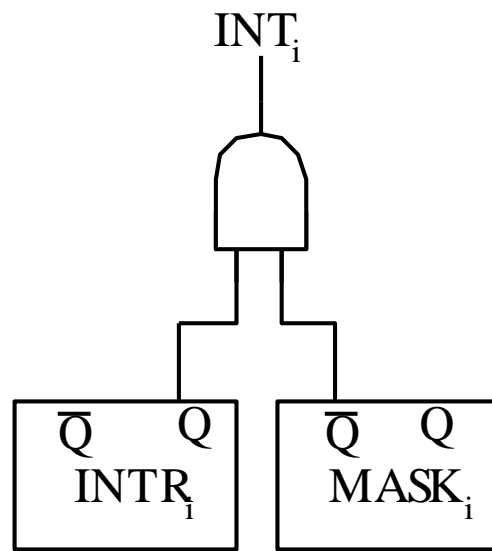
- (1) 当响应某一级中断请求，不再允许被其他中断请求打断时；
- (2) 在中断服务程序的保护和恢复现场之前。

### 3. 中断屏蔽

中断源发出中断请求之后，这个中断请求并不一定能真正送到CPU去，在有些情况下，可以用程序方式有选择地封锁部分中断，这就是中断屏蔽。

## 9.3 中断系统和程序中中断方式

如果给每个中断源都相应地配备一个中断屏蔽触发器MASK, 则每个中断请求信号在送往判优器之前, 还要受到屏蔽触发器的控制。当 $MASK_i=1$ , 表示对应中断源的请求被屏蔽（封锁其中断源的请求），可见中断请求触发器和中断屏蔽触发器是成对出现的，只有当 $INTR_i=1$ （中断源有中断请求）， $MASK_i=0$ （该级中断未被屏蔽），才允许对应的中断请求送往CPU。



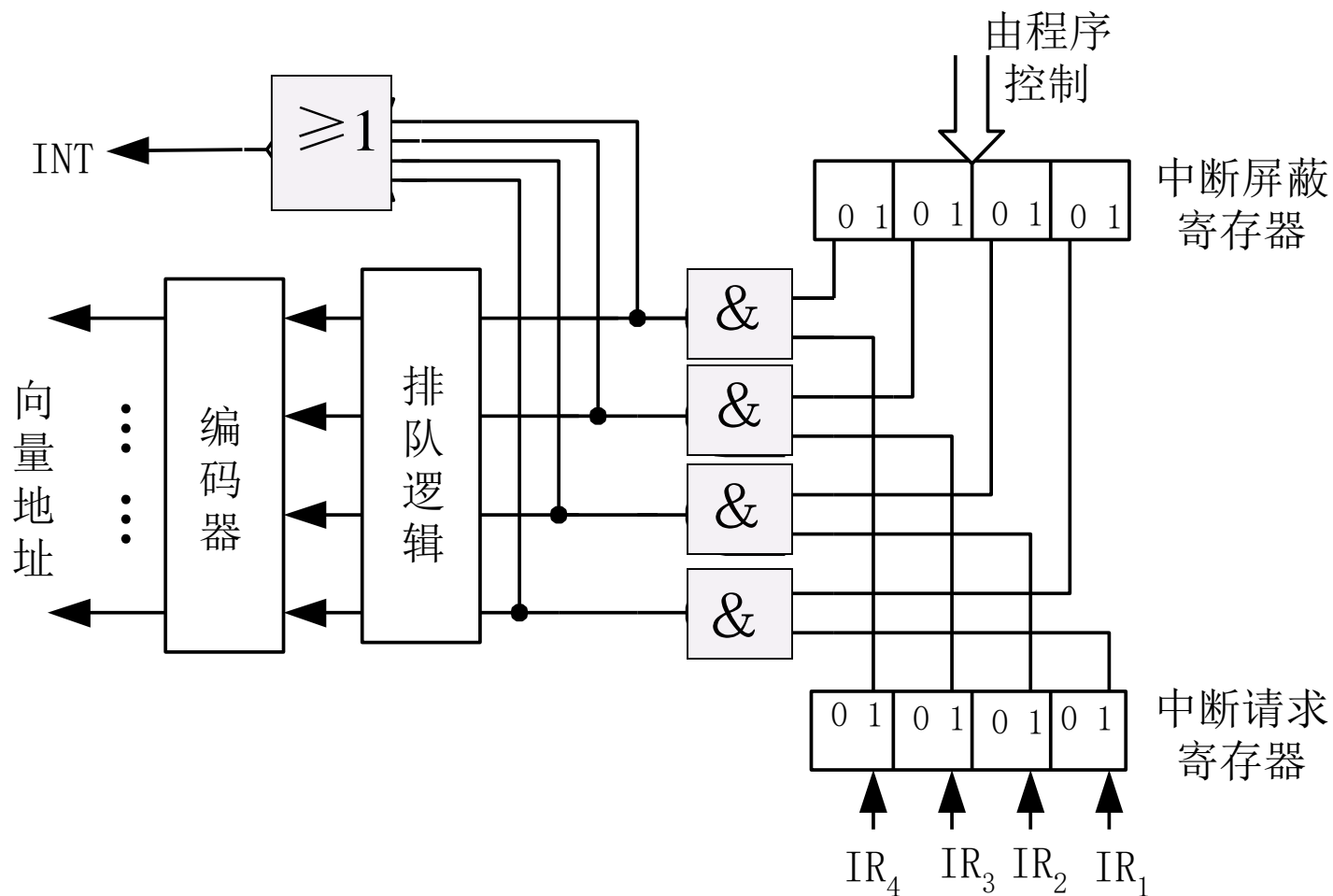
## 9.3 中断系统和程序中中断方式

在中断接口电路中，多个屏蔽触发器组成一个屏蔽寄存器，其内容称为屏蔽字或屏蔽码，由程序来设置。屏蔽字某一位的状态将成为本中断源能否真正发出中断请求信号的必要条件之一。

这样，就可实现CPU对中断处理的控制，使中断能在系统中合理协调地进行。中断屏蔽寄存器的作用：用程序设置的方法将屏蔽寄存器中的某一位置“1”，则对应的中断请求被封锁，无法去参加排队判优；若屏蔽寄存器中的某一位置“0”，才允许对应的中断请求送往CPU。



## 9.3 中断系统和程序中中断方式



## 9.3 中断系统和程序中中断方式

如一个中断系统有16个中断源，每一个中断源按其优先级别赋予一个屏蔽字。“0”表示开放，“1”表示屏蔽。

中断源的优先级	屏蔽字（16位）
1	111...111
2	011...111
3	001...111
⋮	⋮
15	000...011
16	000...001

第1级中断源的优先级别最高，它禁止本级和更低级的中断请求；第16级中断源的优先级别最低，它仅禁止本级的中断请求，而对其他高级的中断请求全部开放。

## 9.3 中断系统和程序中中断方式

### 4. 中断升级

中断屏蔽字的另一个作用是可以改变中断优先级，将原级别较低的中断源变成较高的级别，我们称之为中断升级。这实际上是一种动态改变优先级的方法。

这里所说的改变优先次序是指改变中断的处理次序。中断处理次序和中断响应次序是两个不同的概念，**中断响应次序是由硬件排队电路决定的，无法改变。**但是，**中断处理次序是可以由屏蔽码来改变的**，故把屏蔽码看成软排队器。中断处理次序可以不同于中断响应次序。

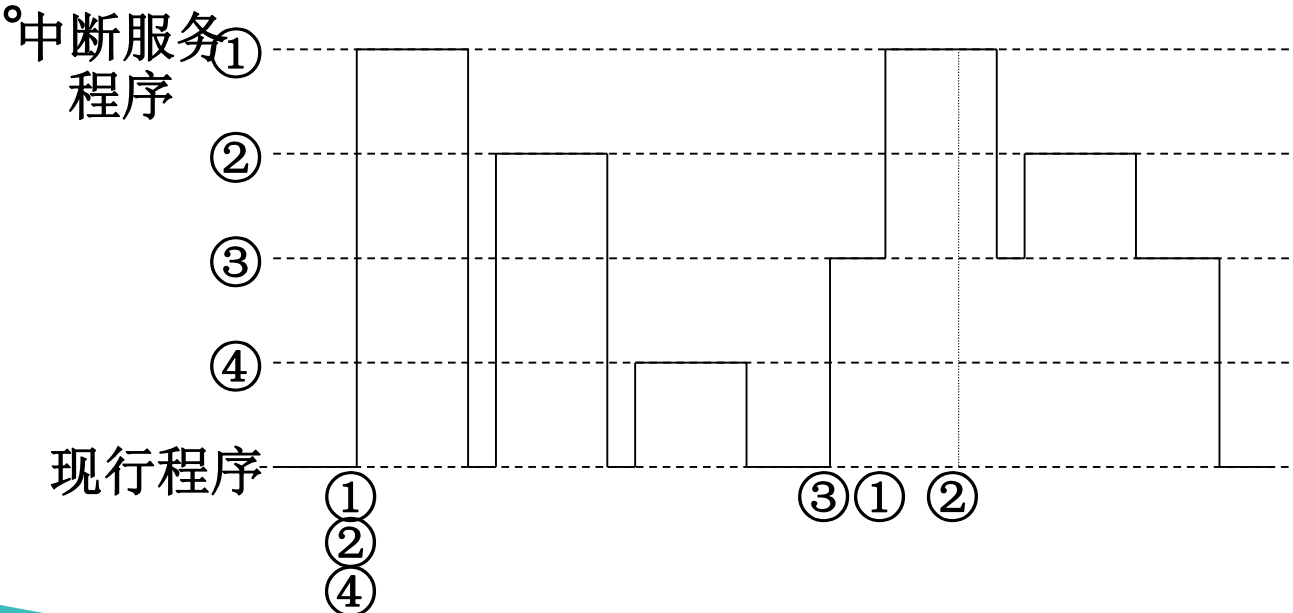
## 9.3 中断系统和程序中中断方式

例如，某计算机的中断系统有4个中断源，每个中断源对应一个屏蔽码。中断响应的优先次序为1→2→3→4。中断的处理次序和中断的响应次序是一致的。

程序级别	屏蔽码			
	1级	2级	3级	4级
第1级	1	1	1	1
第2级	0	1	1	1
第3级	0	0	1	1
第4级	0	0	0	1

## 9.3 中断系统和程序中中断方式

根据这一次序，可以看到CPU运动的轨迹，当多个中断请求同时出现时，处理次序与响应次序一致；当中断请求先后出现时，允许优先级别高的中断请求打断优先级别低的中断服务程序，实现中断嵌套。



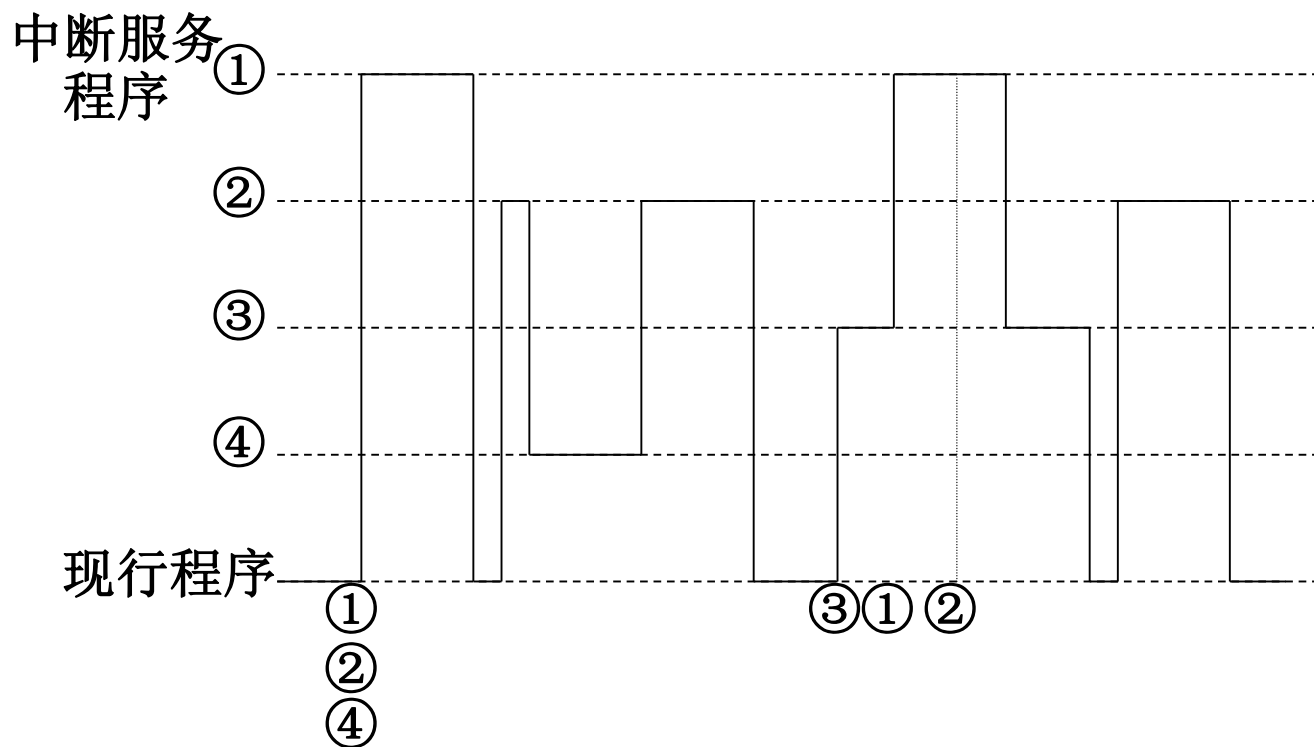
## 9.3 中断系统和程序中中断方式

在不改变的中断响应次序下，通过修改屏蔽码，可使中断处理次序改为1→4→3→2。

程序级别	屏蔽码			
	1级	2级	3级	4级
第1级	1	1	1	1
第2级	0	1	0	0
第3级	0	1	1	0
第4级	0	1	1	1

## 9.3 中断系统和程序中中断方式

在同样中断请求的情况下，CPU的运动轨迹发生了变化。



## 9.3 中断系统和程序中中断方式

### 9.3.5 中断全过程

中断全过程是指从中断源发出中断请求开始，CPU响应这个请求，现行程序被中断，转至中断服务程序，直至中断服务程序执行完毕，CPU再返回原来的程序继续执行的整个过程。

中断全过程分为五个阶段：

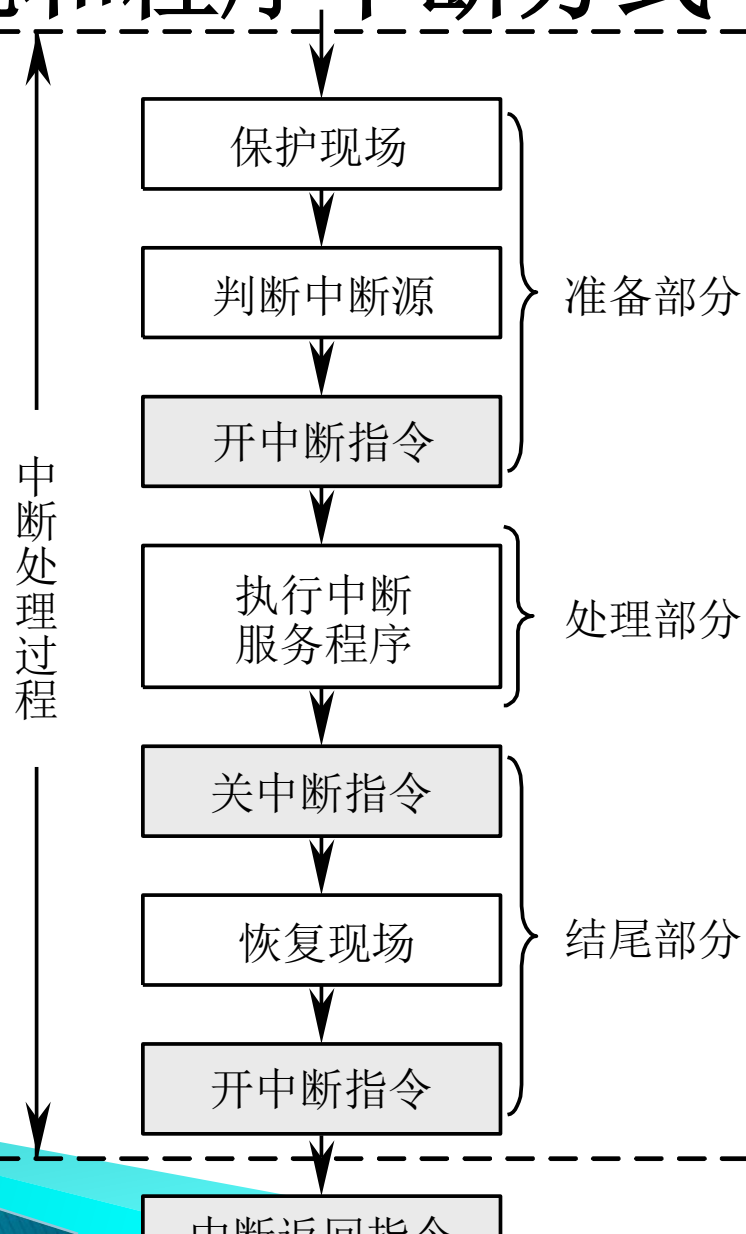
中断请求、中断判优、中断响应、中断处理、中断返回。



## 9.3 中断系统和程序中中断方式

其中中断处理就是执行中断服务程序，中断服务程序基本上由三部分组成，第一部分为准备部分，其基本功能是保护现场，对于非向量中断方式则需要确定中断源，最后开放中断，允许更高级的中断请求打断低级的中断服务程序。第二部分为处理部分，即真正执行为某个中断源服务的中断服务程序。第三部分为结尾部分，首先要关中断，以防止在恢复现场过程中被新的中断打断，接着恢复现场，然后开放中断，以便返回原来的程序后可响应其它的中断请求。

## 9.3 中断系统和程序中中断方式



## 9.4 DMA方式及其接口

### 9.4.1 DMA方式的基本概念

#### 1. DMA方式的特点 ■

无论程序查询还是程序中断方式，主要的工作都是由CPU执行程序完成的，这需要花费时间，因此不能实现高速外设与主机的信息交换。

直接存储器访问DMA方式是在**外设和主存储器之间**开辟一条“**直接数据通道**”，在不需要CPU干预也不需要软件介入的情况下在两者之间进行的高速数据传送方式。

## 9.4 DMA方式及其接口

在DMA传送方式中，对数据传送过程进行控制的硬件称为DMA控制器。当外设需要进行数据传送时，通过DMA控制器向CPU提出DMA传送请求，CPU响应之后将让出系统总线，由DMA控制器接管总线进行数据传送。

DMA方式具有下列特点：

① 它使主存与CPU的固定联系脱钩，主存既可被CPU访问，又可被外设访问。

## 9.4 DMA方式及其接口

② 在数据块传送时，主存地址的确定，传送数据的计数等等都用硬件电路直接实现。

③ 主存中要开辟专用缓冲区，及时供给和接收外设的数据。

④ DMA传送速度快，CPU和外设并行工作，提高了系统的效率。

⑤ DMA在开始前和结束后要通过程序和中断方式进行预处理和后处理。

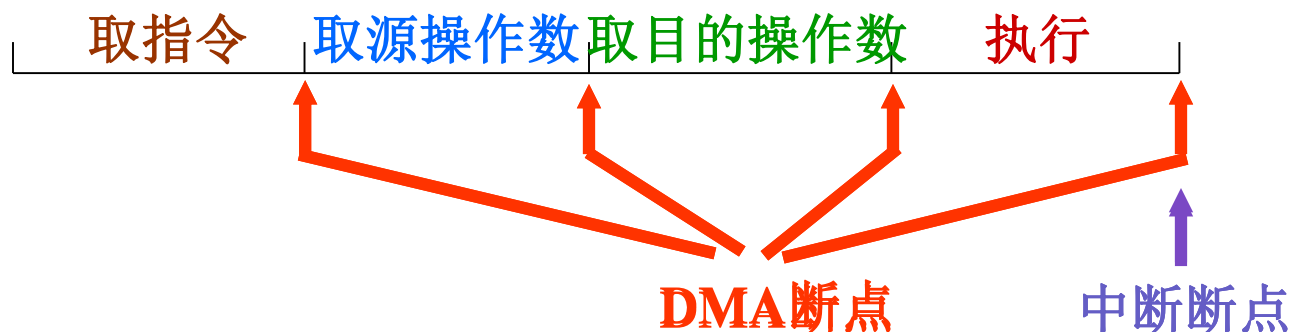
## 9.4 DMA方式及其接口

### 2. DMA和中断的区别 ■

两者的重要区别为： ■

① 中断方式是程序切换，需要保护和恢复现场；而DMA方式除了开始和结尾时，不占用CPU的任何资源。

② 对中断请求的响应只能发生在每条指令执行完毕时；而对DMA请求的响应可以发生在每个机器周期结束时。



## 9.4 DMA方式及其接口

③ 中断传送过程需要CPU的干预；而DMA传送过程不需要CPU的干预，故数据传送速率非常高，适合于高速外设的成组数据传送。

④ DMA请求的优先级高于中断请求。

⑤ 中断方式具有对异常事件的处理能力；而DMA方式仅局限于完成传送信息块的I/O操作。

## 9.4 DMA方式及其接口

### 9.4.2 DMA接口（DMA控制器）

#### 1.DMA控制器的功能

在DMA传送过程中，DMA控制器将接管CPU的地址总线、数据总线和控制总线，CPU的主存控制信号被禁止使用。而当DMA传送结束后，将恢复CPU的一切权利并开始执行其操作。由此可见，DMA控制器必须具有控制系统总线的能力，即能够像CPU一样输出地址信号，接收或发出控制信号，输入或输出数据信号。



## 9.4 DMA方式及其接口

**DMA**控制器在外设与主存之间直接传送数据期间，完全代替**CPU**进行工作，它的主要功能有：

- (1) 接受外设发出的**DMA**请求，并向**CPU**发出总线请求；
- (2) 当**CPU**响应此总线请求，发出总线响应信号后，接管对总线的控制，进入**DMA**操作周期；
- (3) 确定传送数据的主存单元地址及传送长度，并能自动修改主存地址计数值和传送长度计数值；

## 9.4 DMA方式及其接口

(4) 规定数据在主存与外设之间的传送方向，发出读/写或其他控制信号，并执行数据传送的操作。

(5) 向CPU报告DMA操作的结束。

## 9.4 DMA方式及其接口

### 2. DMA控制器的基本组成 ■

#### (1) 主存地址计数器

用来存放主存中要交换数据的地址，该计数器的初始值为主存缓冲区的首地址，当DMA传送时，每传送一个数据，将地址计数器加“1”，从而以增量方式给出主存中要交换的一批数据的地址，直至这批数据传送完毕为止。

## 9.4 DMA方式及其接口

### (2) 传送长度计数器

用来记录传送数据块的长度，其初始值为传送数据的总字数或总字节数，每传送一个字或一个字节，计数器自动减“1”，当其内容为“0”时表示数据已全部传送完毕。

### (3) 数据缓冲寄存器

用来暂存每次传送的数据。输入时，数据由外设（如磁盘）先送往数据缓冲寄存器，再通过数据总线送到主存。反之，输出时，数据由主存通过数据总线送到数据缓冲寄存器，然后再送到外设。

## 9.4 DMA方式及其接口

### (4) DMA请求触发器

每当外设准备好一个数据后给出一个控制信号，使DMA请求触发器置位，控制/状态逻辑经系统总线向CPU发出总线请求（HOLD），如果CPU响应，发回批准信号（HLDA），DMA控制器接管总线控制权，向系统总线送出传送命令与总线地址。控制/状态逻辑接收此信号后使DMA请求触发器复位，为交换下一个数据做准备。

## 9.4 DMA方式及其接口

### (5) 控制/状态逻辑

它由控制和时序电路以及状态标志等组成，用于指定传送方向，修改传送参数，并对DMA请求信号和CPU响应信号进行协调和同步。

### (6) 中断机构

当一个数据块传送完毕，由溢出信号触发中断机构，向CPU提出中断请求，CPU将进行DMA传送的结尾处理。

# 9.4 DMA方式及其接口

## 3.DMA控制器的引出线

### (1)地址总线

在DMA方式下，呈输出状态，可对主存进行地址选择；在CPU方式下，呈输入状态，可对DMA控制器中的有关寄存器进行寻址。

### (2)数据总线

在DMA方式下，用它进行数据传送；在CPU方式下，可对DMA控制器的有关寄存器进行编程。

## 9.4 DMA方式及其接口

### (3)控制数据传送方式的信号线

存储器读信号、存储器写信号、外设读信号、  
外设写信号。

### (4)DMA控制器与外设之间的联络信号线

DMA请求信号

DMA响应信号

### (5)DMA控制器与CPU之间的联络信号线

总线请求

总线响应信号



## 9.4 DMA方式及其接口

### 4.DMA控制器的连接和传送

- (1) 首先由外设向DMA控制器发出请求信号DREQ。
- (2) DMA控制器向CPU发出总线请求信号HRQ。
- (3) CPU向DMA控制器发出总线响应信号HLDA，此时，DMA控制器获取了总线的控制权。
- (4) DMA控制器向外设发出DMA响应信号DACK，表示DMA控制器已控制了总线，允许外设与主存交换数据。

## 9.4 DMA方式及其接口

(5) **DMA**控制器按主存地址计数器的内容发出地址信号作为主存地址的选择，同时主存地址计数器的内容加1（或减1）。

(6) **DMA**控制器发出**IOR**信号到外设，将外设数据读入总线，同时发出**MEMW**信号，将数据总线的数写入地址总线选中的主存单元。

(7) 传送长度计数器减 1 。

重复(5)(6)(7)步骤，直到字节计数器减到“0”为止，数据块的**DMA**方式传送工作宣告完成。这时，**DMA**控制器的**HRQ**降为低电平，总线控制权交还CPU。

# 9.4 DMA方式及其接口

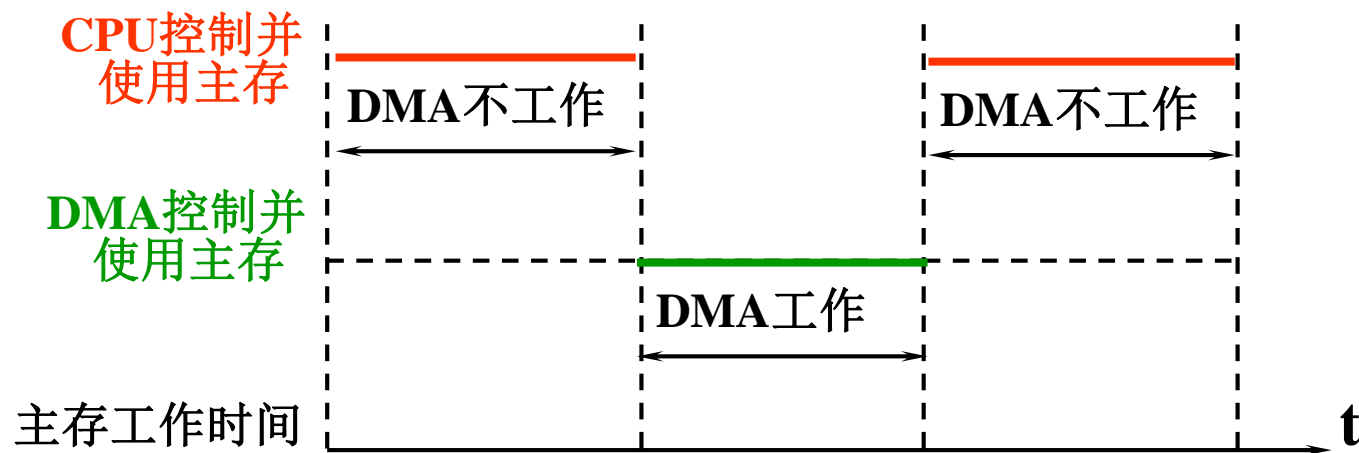
## 9.4.3 DMA传送方法与传送过程

### 1. DMA传送方法

#### (1) CPU停止访问主存法

用DMA请求信号迫使CPU让出总线控制权，CPU在现行机器周期执行完成之后，使其数据、地址总线处于三态，并输出总线批准信号。每次DMA请求获得批准后，DMA控制器获得总线控制权以后，连续占用若干个存取周期（总线周期）进行成组连续的数据传送，直至批量传送结束，DMA控制器才把总线控制权交回CPU。在DMA操作期间，CPU处于保持状态，停止访问主存，仅能进行一些与总线无关的内部操作。这种方法只适用于高速外设的成组传送。

## 9.4 DMA方式及其接口



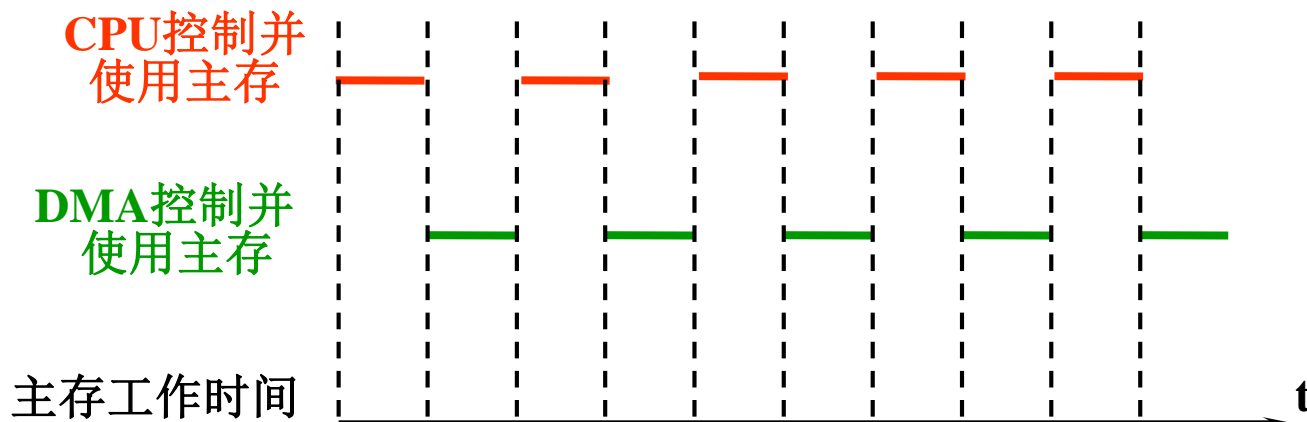
## 9.4 DMA方式及其接口

### (2) 存储器分时法

把原来的一个存取周期分成两个时间片，一片给CPU，一片给DMA，使CPU和DMA交替地访问主存。这种方法不需要申请和归还总线，使总线控制权的转移几乎不需要什么时间，所以对DMA传送来讲效率是很高的，而且CPU既不停止现行程序的运行，也不进入保持状态，在CPU不知不觉中便进行了DMA传送，但这种方法需要主存在原来的存取周期内为两个部件服务，如果要维持CPU的访存速度不变，就要求主存的工作速度提高一倍。

## 9.4 DMA方式及其接口

另外，由于大多数外设的速度都不能与CPU相匹配，所以供DMA使用的时间片可能成为空操作，将会造成一些不必要的浪费。

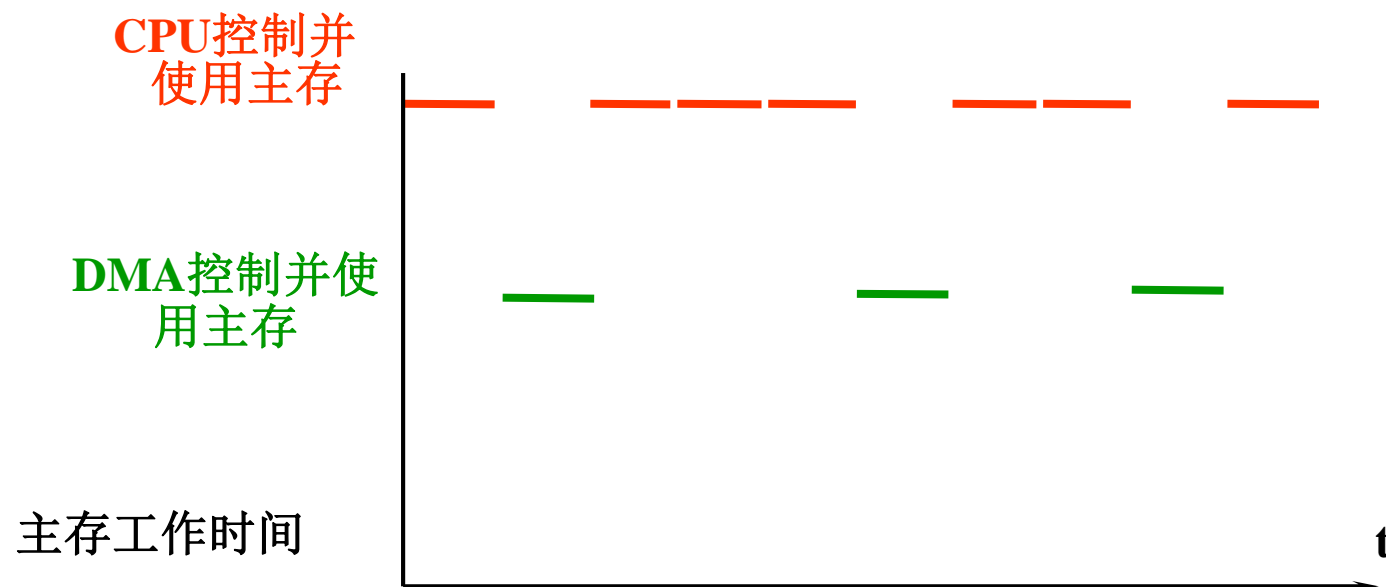


## 9.4 DMA方式及其接口

### (3) 周期挪用法

周期挪用法是前两种方法的折衷。一旦外设有DMA请求并获得CPU批准后，CPU让出一个周期的总线控制权，由DMA控制器控制系统总线，挪用一個存取周期进行一次数据传送，传送一个字节或一个字，然后，DMA控制器将总线控制权交回CPU，CPU继续进行自己的操作，等待下一个DMA请求的到来；重复上述过程，直至数据块传送完毕。如果在同一时刻，发生CPU与DMA的访存冲突，那么优先保证DMA工作，而CPU等待一个存取周期。若DMA传送时CPU不需要访存，则外设的周期挪用对CPU执行程序无任何影响。

## 9.4 DMA方式及其接口





# 9.4 DMA方式及其接口

## 2. DMA传送过程

### (1) DMA预处理

这是在DMA传送之前做的一些必要的准备工作，是由CPU来完成的。CPU首先执行几条I/O指令，用于测试外设的状态、向DMA控制器的有关寄存器置初值、设置传送方向、启动该外部设备等。

在这些工作完成之后，CPU继续执行原来的程序，在外设准备好发送的数据（输入时）或接收的数据已处理完毕（输出时），外设向DMA控制器发DMA请求，再由DMA控制器向CPU发总线请求。

## 9.4 DMA方式及其接口

### (2) 数据传送

DMA的数据传送可以是以单字节（或字）为基本单位，也可以以数据块为基本单位。对于以数据块为单位的传送，DMA 占用总线后的数据输入和输出操作都是通过循环来实现的。

需要特别指出的是，这一循环不是由CPU执行程序实现的，而是由DMA控制器实现的。

## 9.4 DMA方式及其接口

### (3) DMA后处理

当长度计数器计为0时，DMA操作结束，DMA控制器向CPU发中断请求，CPU 停止原来程序的执行，转去执行中断服务程序做DMA结束处理工作。

## 9.5 通道控制方式 ■

### 9.5.1 通道的基本概念

在大型计算机系统中，所连接的I/O设备数量多，输入/输出频繁，要求整体的速度快，单纯依靠主CPU采取中断和DMA等控制方式已不能满足要求。

#### 1. 通道控制方式与DMA方式的区别 ■

通道控制方式是DMA方式的进一步发展，实质上，通道也是实现外设和主存之间直接交换数据的控制器。两者的主要区别在于：

## 9.5 通道控制方式

① **DMA**控制器是通过专门设计的硬件控制逻辑来实现对数据传送的控制；而通道则是一个具有特殊功能的处理器，它具有自己的指令和程序，通过执行一个通道程序实现对数据传送的控制，故通道具有更强的独立处理数据输入/输出的功能。

② **DMA**控制器通常只能控制一台或少数几台同类设备；而一个通道则可以同时控制许多台同类或不同类的设备。

# 9.5 通道控制方式

## 2. 通道的功能

通道在一定的硬件基础上利用软件手段实现对I/O的控制和传送，更多地免去了CPU的介入，从而使主机和外设的并行工作程度更高。当然，通道并不能完全脱离CPU，它还要受到CPU的管理，而且通道还应该向CPU报告自己的状态，以便CPU决定下一步的处理。通道的功能：

① 接受CPU的I/O指令，按指令要求与指定的外设进行联系。

## 9.5 通道控制方式

② 从主存取出属于该通道程序的通道指令，经译码后向设备控制器和设备发送各种命令。

③ 实施主存和外设间的数据传送。

④ 从外设获得设备的状态信息，形成并保存通道本身的状态信息，根据要求将这些状态信息送到主存的指定单元，供CPU使用。

⑤ 将外设的中断请求和通道本身的中断请求按次序及时报告CPU。

# 第8章 小结

## 9.1 主机与外设的连接

- ▶ 输入/输出接口
- ▶ 接口的基本组成

接口，端口

## 9.2 程序查询方式及其接口

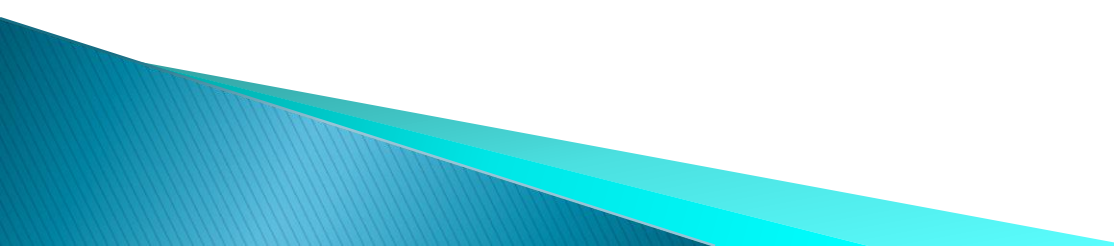
- ▶ 程序查询方式的工作流程

## 9.3 中断系统和程序中断方式

- ▶ 中断的基本概念
- ▶ 中断与调用子程序的区别



# 第8章 小结

- ▶ 中断的基本类型
  - ▶ CPU响应中断的条件
  - ▶ 中断隐指令
  - ▶ 进入中断服务程序（向量地址的形成）
  - ▶ 中断现场的保护和恢复
  - ▶ 允许和禁止中断
  - ▶ 中断屏蔽
- 

# 第8章 小结

## 9.4 DMA方式及其接口

- ▶ **DMA方式的特点**

- ▶ **DMA和中断的区别**

- ▶ **DMA接口**

- ▶ **DMA传送方法**

- ▶ **DMA传送过程**

## 9.5 通道控制方式

- ▶ **通道控制方式与DMA方式的区别**