计算机组成与体系结构

2023

6.1 输入输出系统概述

6.2 磁盘阵列

6.3 总线

6.4 输入输出系统控制方式

6.1 输入输出系统概述

- 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。
- 输入输出系统的主要功能是对指定的外设进行输入、输出操作，同时也完成许多其他的管理和控制。有的输入输出系统还能对要传送的信息进行格式变换，形成和产生有关输入输出操作是否完成或在执行过程中是否有错的状态控制信息，经中断系统传送给操作系统去分析和处理。

6.1 输入输出系统概述

- 输入输出系统的特点
  - 输入输出系统涉及到机、光、电、磁、声、自动控制等多种学科。
  - 用户无需了解输入输出系统和输入输出设备的具体细节就能使用输入输出设备
  - 处理机的外部世界包括：本地和远程用户、系统操作员、操作控制台、输入输出设备、辅助存储器、其它处理机、各种通信设备和虚拟现实系统等。

6.1 输入输出系统概述

- 输入输出系统的特点
  - 1. 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。
  - 2.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。

- 输入输出系统的特点
  - 3.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。
    - 计算机系统的使用者，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。

6.1 输入输出系统概述

- 输入输出系统的组织方式（自学）
  - 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。

6.1 输入输出系统概述

6.2 磁盘阵列

6.3 总线

6.4 输入输出系统控制方式

6.2.1 RAID简介

- RAID是Redundent Array of Inexpensive Disks的缩写，直译为“廉价冗余磁盘阵列”，也简称为“磁盘阵列”。后来RAID中的字母I被改作为Independent，RAID就成了“独立冗余磁盘阵列”，但这只是名称的变化，实质性的内容并没有改变。可以把RAID理解成一种使用磁盘驱动器的方法，它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。一般情况下，组成的逻辑磁盘驱动器的容量要小于各个磁盘驱动器容量的总和。

6.2.1 RAID简介

- RAID的优点
  - 成本低，功耗小，传输速率高。很多磁盘驱动器同时传输数据，而这些磁盘驱动器在逻辑上又是一个磁盘驱动器，所以使用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。
  - 提供容错功能。这是使用RAID的第二个原因，因为如果不考虑磁盘上的循环冗余校验（CRC）码的话，普通磁盘驱动器无法提供容错功能。RAID的容错是建立在每个磁盘驱动器的硬件容错功能之上的，所以它提供更高的安全性。
  - RAID比起传统的大直径磁盘驱动器来，在同样的容量下，价格要低许多。

6.2.2 RAID的分级（自学）

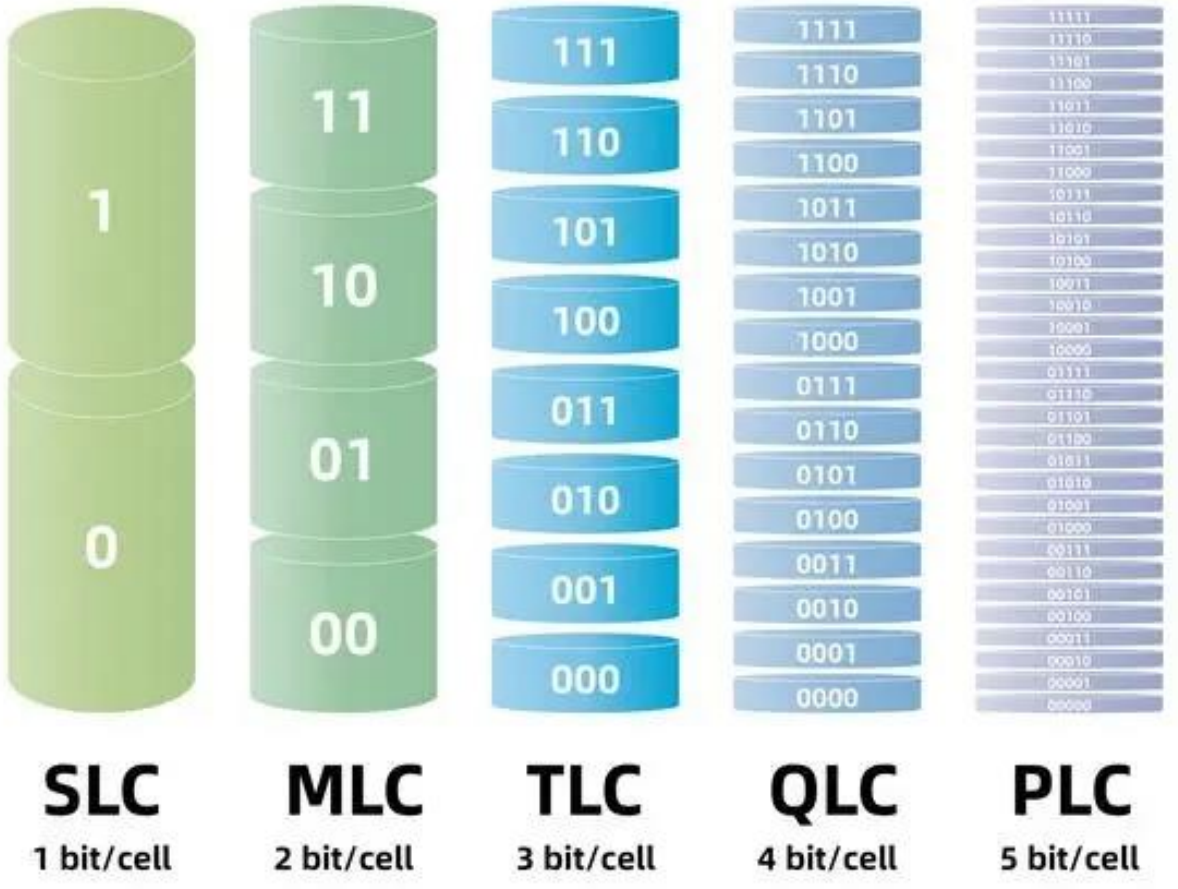
RAID级别	名称	数据磁盘数	可正常工作的最多失效磁盘数	检测磁盘数
RAID0	无冗余无校验的磁盘阵列	8	0	0
RAID1	镜象磁盘阵列	8	1	8
RAID2	纠错海明码磁盘阵列	8	1	4
RAID3	位交叉奇偶校验的磁盘阵列	8	1	1
RAID4	块交叉奇偶校验的磁盘阵列	8	1	1
RAID5	无独立校验盘的奇偶校验磁盘阵列	8	1	1
RAID6	双维无独立校验盘的奇偶校验磁盘阵列	8	2	2

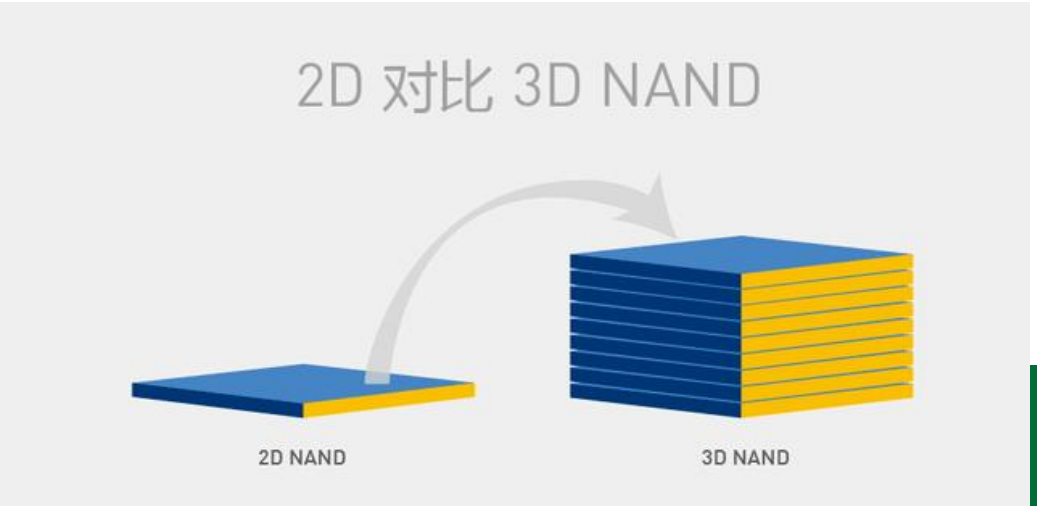
6.2.3 固态硬盘SSD

- SSD定义及组成
  - SSD（Solid State Disk）俗称固态硬盘，相对原来主轴旋转，并无主轴旋转，并无机械部分，所以被人称为固态硬盘
  - SSD由控制单元和存储单元（FLASH芯片）组成，存储单元负责存储资料，控制单元负责读取、写入资料。简单的说就是存储芯片通过阵列制成的硬盘（基本都是RAID 0模式，这也是SSD高速的原因）。固态硬盘的接口规范和定义、功能及使用方法上与普通硬盘的完全相同

SSD基本存储单元

- SLC：Single Level Cell。
- MLC：Multi Level Cell。
- TLC：Trinary Level Cell。
- QLC：Quad Level Cell。
- PLC： Penta-level cell。





6.2 磁盘阵列

6.2.3 固态硬盘SSD

6.2.3 固态硬盘SSD

- 优点
  - 读写速度快：采用闪存作为存储介质，读取速度相对机械硬盘更快。固态硬盘不用磁头，寻道时间几乎为0。
  - 低功耗：固态硬盘的功耗上要低于传统硬盘。
  - 无噪音：固态硬盘没有机械马达和风扇，工作时噪音值为0分贝。
  - 防震抗摔性：传统硬盘都是磁碟型的，数据储存在磁碟扇区里。而固态硬盘是使用闪存颗粒（即MP3、U盘等存储介质）制作而成，所以SSD固态硬盘内部不存在任何机械部件，这样即使在高速移动甚至伴随翻转倾斜的情况下也不会影响到正常使用，而且在发生碰撞和震荡时能够将数据丢失的可能性降到最小。

6.2.3 固态硬盘SSD

- 优点
  - 工作温度范围大：典型的硬盘驱动器只能在5到55摄氏度范围内工作。而大多数固态硬盘可在-10~70摄氏度工作。
  - 轻便：固态硬盘在重量方面更轻，与常规1.8英寸硬盘相比，重量轻20-30克。
- 缺点
  - 容量：截止2021年1月世界上容量最大的固态硬盘是Nimbus Data推出的 ExaDrive DC100 系列固态硬盘，容量可达100TB。
  - 售价高：截止2021年1月市场上每GB大约0.6-1元。相比每GB仅为0.2元的机械硬盘高了不少。

6.2.3 固态硬盘SSD

- 缺点
  - 寿命限制：固态硬盘闪存具有擦写次数限制的问题，这也是许多人诟病其寿命短的所在。闪存完全擦写一次叫做1次P/E，因此闪存的寿命就以P/E作单位。34nm的闪存芯片寿命约是5000次P/E，而25nm的寿命约是3000次P/E。某些应用，如操作系统的LOG记录等，可能会对某一扇区进行多次反复读写，而这种情况下，固态硬盘的实际寿命还未经考验。

6.2.3 固态硬盘SSD

- 缺点
  - 寿命限制：另外，虽然固态硬盘的每个扇区可以重复擦写100000次(SLC)，但某些应用，如操作系统的LOG记录等，可能会对某一扇区进行多次反复读写，而这种情况下，固态硬盘的实际寿命还未经考验。不过通过均衡算法对存储单元的管理，其预期寿命会延长。SLC有10万次的写入寿命，成本较低的MLC，写入寿命仅有1万次,而廉价的TLC闪存则更是只有1000-2000次。此外，使用全盘模拟SLC提升写入速度的多阶存储固态会面临写入放大问题，进一步缩短寿命。

6.1 输入输出系统概述

6.2 磁盘阵列

6.3 总线

6.4 输入输出系统控制方式

6.3 总线

- 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。

6.3.1 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。总线的优点是成本低、简单；缺点是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。

6.3.1 总线特点

1. 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。

6.3.1 总线特点

2.总线使用权

- 总线是由多个部件和设备所共享的，为了正确地实现它们之间的通信，必须有一个总线控制机构，对总线的使用进行合理的分配和管理。
- 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。

6.3.2总线的数据宽度与总线线数

1. 数据宽度

- 数据宽度是I/O设备取得I/O总线后所传送数据的总量，它不同与前述的数据通路宽度。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。而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。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。

6.3.2总线的数据宽度与总线线数

1. 数据宽度

- 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。
- 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。

6.3.2总线的数据宽度与总线线数

1. 数据宽度

- 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。不过，若传送的信息块小于定长块的大小，但字数又不少时，设备或总线的利用率会降低。
- 单字加变长块宽度是一种灵活有效但却是复杂、花钱的方法。当要求传送单字时比只能成块传送的方法节省了不少起始辅助操作；而当成块传送时，块的大小又能调整到与部件和应用的要求相适应，从而优化了总线的使用。

6.3.2总线的数据宽度与总线线数

2. 总线的线数

- 总线需要有发送电路、接收电路、传输导线或电缆、转接插头和电源等，这部分比起逻辑线路的成本高得多，而且转接器往往占系统物理空间的相当部分，是降低系统可靠性的主要部分。总线的线数越多，成本越高、干扰越大、可靠性越低、占用的空间也越大，当然传送速度和流量也越高。 此外，总线的长度越长，成本越高，干扰越大，波形畸变越严重，可靠性越低。为此，越是长的总线，其线数就应尽可能减少。数据总线的宽度有一位、一个字节或一个全字等等。

6.3.2总线的数据宽度与总线线数

2. 总线的线数

- 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。

6.3.3总线的性能指标

1. 总线宽度

- 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。
  - 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。

6.3.3 总线的性能指标

2. 总线带宽

- 总线带宽定义为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。

6.3.3 总线的性能指标

2. 总线带宽

- 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*=33MHz=33×10⁶/s，*n*=32位，根据定义可得
- *Dr*=4×33×10⁶/s=132MB/s

6.3.3 总线的性能指标

2. 总线带宽

口例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=10MHz，时钟周期 T=1/f=0.1μs，总线周期=2T=0.2μs。一个总线周期中并行传输4字节信息，则总线带宽是4÷0.2=20MB/s。

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。

6.3.3 总线的性能指标

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。

5. 总线猝发传输

- 猝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。

6.3.4 总线定时控制

- 总线的定时控制方式一般分为同步方式和异步方式。

6.3.5 总线的控制方式

- 1. 总线的集中仲裁方式
  - 为了保证同一时刻只有一个申请者使用总线，总线控制机构中设置有总线判优和仲裁控制逻辑，即按照一定的优先次序来决定哪个部件首先使用总线，只有获得总线使用权的部件，才能开始数据传送。总线控制逻辑集中在一处（如在CPU中）的，称为集中式控制，就集中式控制而言，有3种常见的优先权仲裁方式：

6.3.5总线的控制方式

- 1. 总线的集中仲裁方式
  - (1)链式查询方式
    - 链式查询方式的总线控制器使用三根控制线与所有部件和设备相连:
    - 总线请求（BR）：该线有效，表示至少有一个部件或设备要求使用总线。
    - 总线忙（BS）：该线有效，表示总线正在被某部件或设备使用。
    - 总线批准（BG）：该线有效，表示总线控制器响应总线请求。

6.3.5总线的控制方式

- 1. 总线的集中仲裁方式
  - (1)链式查询方式

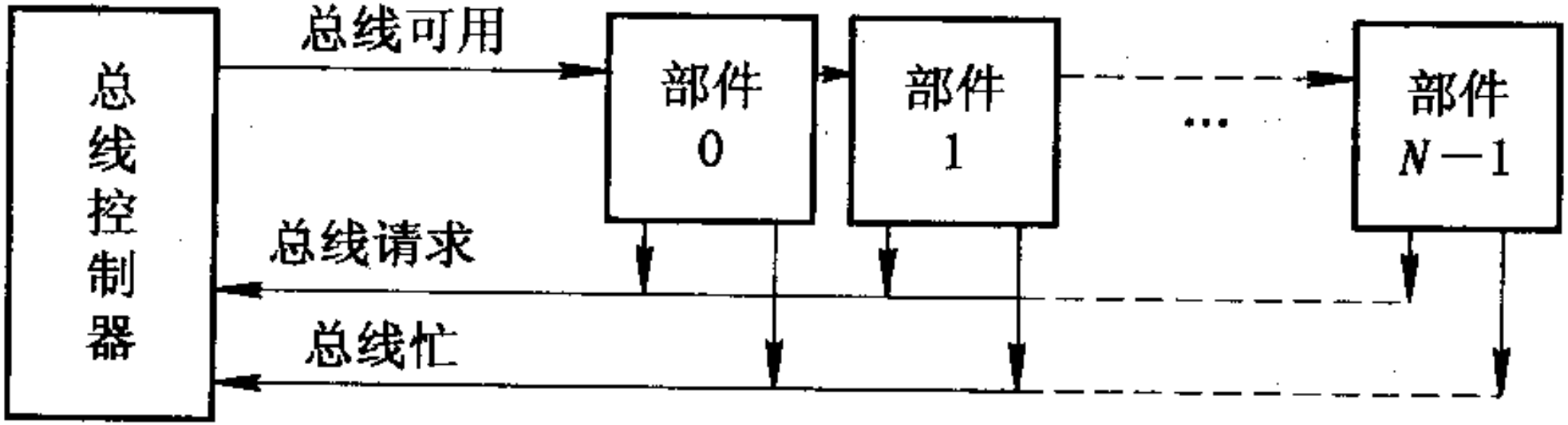


图 6.2 集中式串行链接

6.3.5总线的控制方式

- 1. 总线的集中仲裁方式
  - (1)链式查询方式
    - 链式查询的优点是只用3根线就能按一定的优先次序来实现总线控制，并很容易扩充。缺点是对查询链的故障很敏感，如果第*i*个部件中的查询链电路有故障，那么第*i*个以后的部件都不能工作。另外，因为查询的优先级是固定的，所以当优先级较高的部件出现频繁的总线请求时，优先级较低的部件就可能会难以得到响应。

6.3.5总线的控制方式

- 1. 总线的集中仲裁方式
  - ➢(2)计数器定时查询方式
    - 口计数定时查询方式的总线上的每个部件可以通过公共的BR 线发出请求，总线控制器收到请求之后，在BS为“0”的情况下， 让计数器开始计数，定时地查询各个部件以确定是谁发出的请求。当查询线上的计数值与发出请求的部件号一致时，该部件就使BS线置“1”，获得了总线使用权，并中止计数查询，直至该部件完成数据传送之后，撤消BS信号。

6.3.5总线的控制方式

- 1. 总线的集中仲裁方式
  - ➢(2)计数器定时查询方式

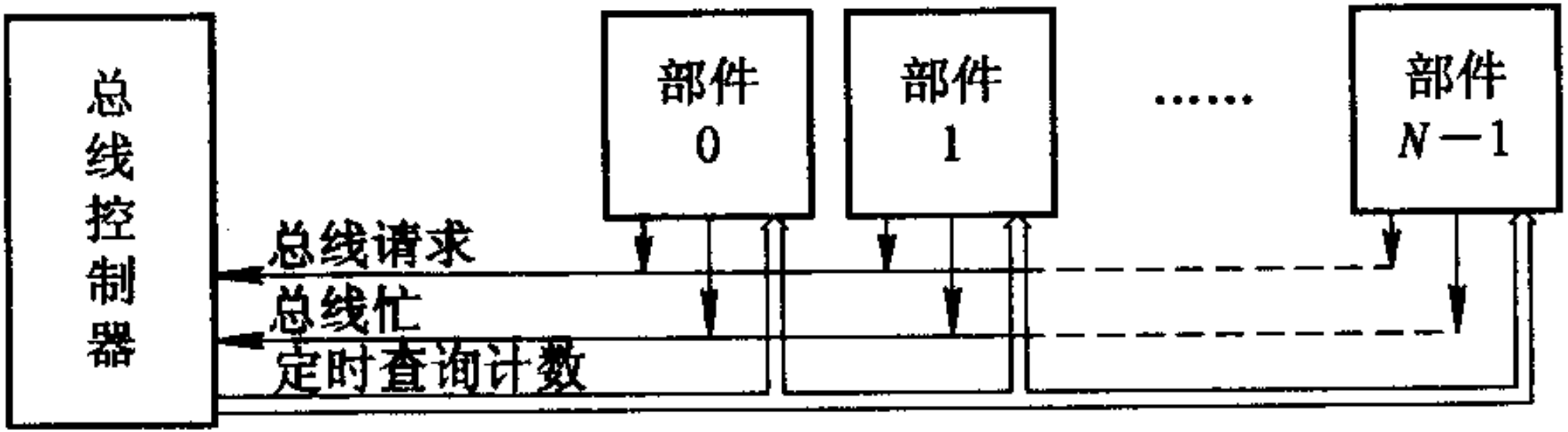


图 6.3 集中式定时查询

6.3.5总线的控制方式

- 1. 总线的集中仲裁方式
  - ➢(2)计数器定时查询方式
    - 口这种计数可以从“0”开始，也可以从中止点开始。如果从“0”开始，各部件的优先次序和链式查询方式相同，优先级的次序是固定的。如果从中止点开始，即为循环优先级，各个部件使用总线的级别将相等。计数器的初始值还可以由程序来设置，这就可以方便地改变优先次序，增加系统的灵活性。定时查询方式的控制线数较多，对于n个部件，共需$2+[log_2 n]$根线。

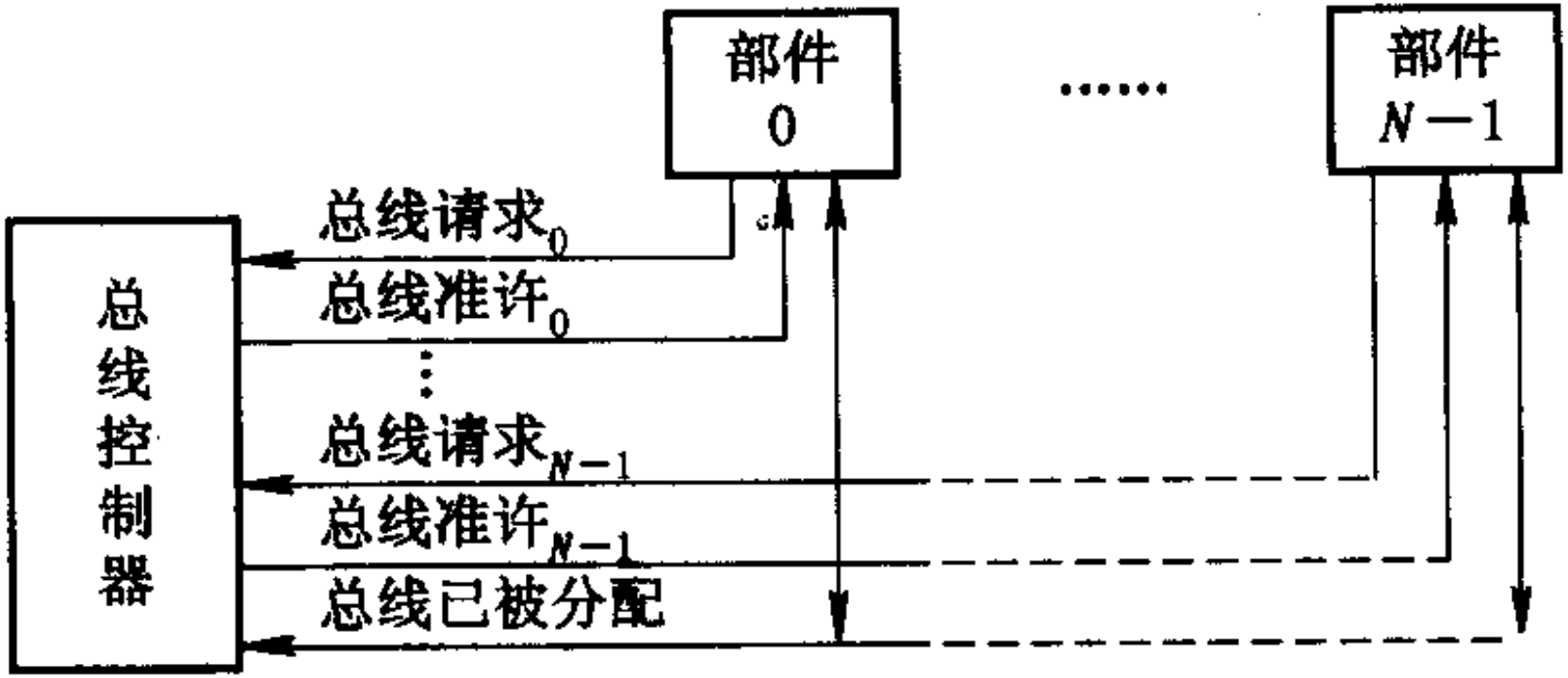
6.3.5总线的控制方式

- 1. 总线的集中仲裁方式
  - (3)独立请求方式
    - 在独立请求方式中， 每一个共享总线的部件均有一对控制线：总线请求BRi和总线批准BGi。当某个部件请求使用总线时，便发出BRi，总线控制器中有一排队电路，根据一定的优先次序决定首先响应哪个部件的请求BRi，然后给该部件送回批准信号BGi。
    - 独立请求方式的优点是响应时间快，然而这是以增加控制线数和硬件电路为代价的。对于n个部件，控制线的数目将达$2n+1$根。 此方式对优先次序的控制也是相当灵活的，它可以预先固定，也可以通过程序来改变优先次序。

6.3.5总线的控制方式

- 1. 总线的集中仲裁方式
  - ➢(3)独立请求方式

图 6.4 集中式独立请求



6.3.5总线的控制方式

- 2. 总线的分布仲裁方式
  - 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

6.3.5总线的控制方式

- 2. 总线的分布仲裁方式
  - （1）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。

6.3.5总线的控制方式

- 2. 总线的分布仲裁方式
  - （2）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就是使用该方案。

6.3.5总线的控制方式

- 2. 总线的分布仲裁方式
  - (3) 并行竞争分布式
    - 这是一种较复杂但有效的裁决方案。其基本思想是：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。

6.1 输入输出系统概述

6.2 磁盘阵列

6.3 总线

6.4 输入输出系统控制方式

6.4 输入输出系统控制方式

- 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。
- 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

6.4 输入输出系统控制方式

- 输入/输出信息传送控制方式
- 2. 程序中断方式
  - 外设在作好输入/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。

6.4 输入输出系统控制方式

- 输入/输出信息传送控制方式
- 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。

6.4 输入输出系统控制方式

- 输入/输出信息传送控制方式
- 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）。

6.4.1 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。
  - ➢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。

6.4.1 程序查询方式

- 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。
  - （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。

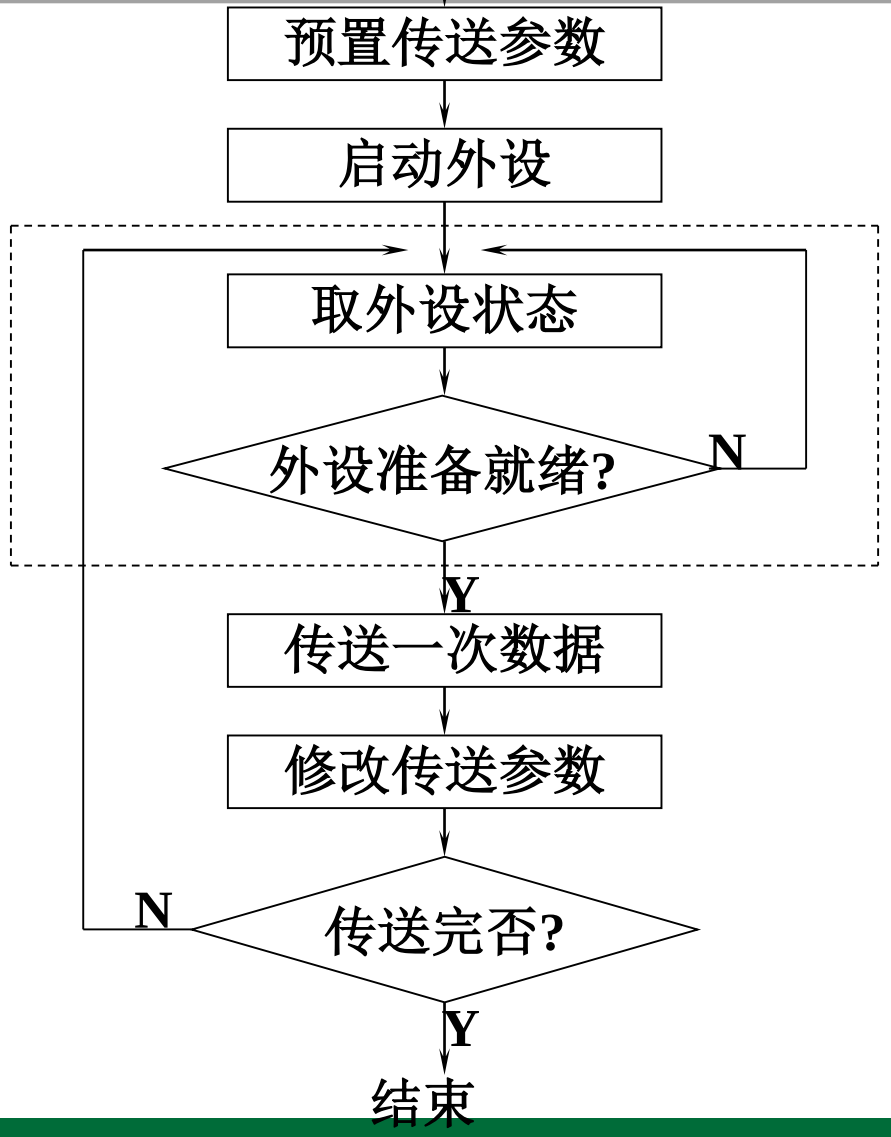
6.4.1 程序查询方式

- 2. 程序查询方式的工作流程
  - ➢ （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。
  - ➢ （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。

6.4.1 程序查询方式

- 2. 程序查询方式的工作流程
  - （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。
  - （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。

6.4 输入输出系统控制方式



输出指令(OUT 控制口,AL)

输入指令(IN AL,状态口)

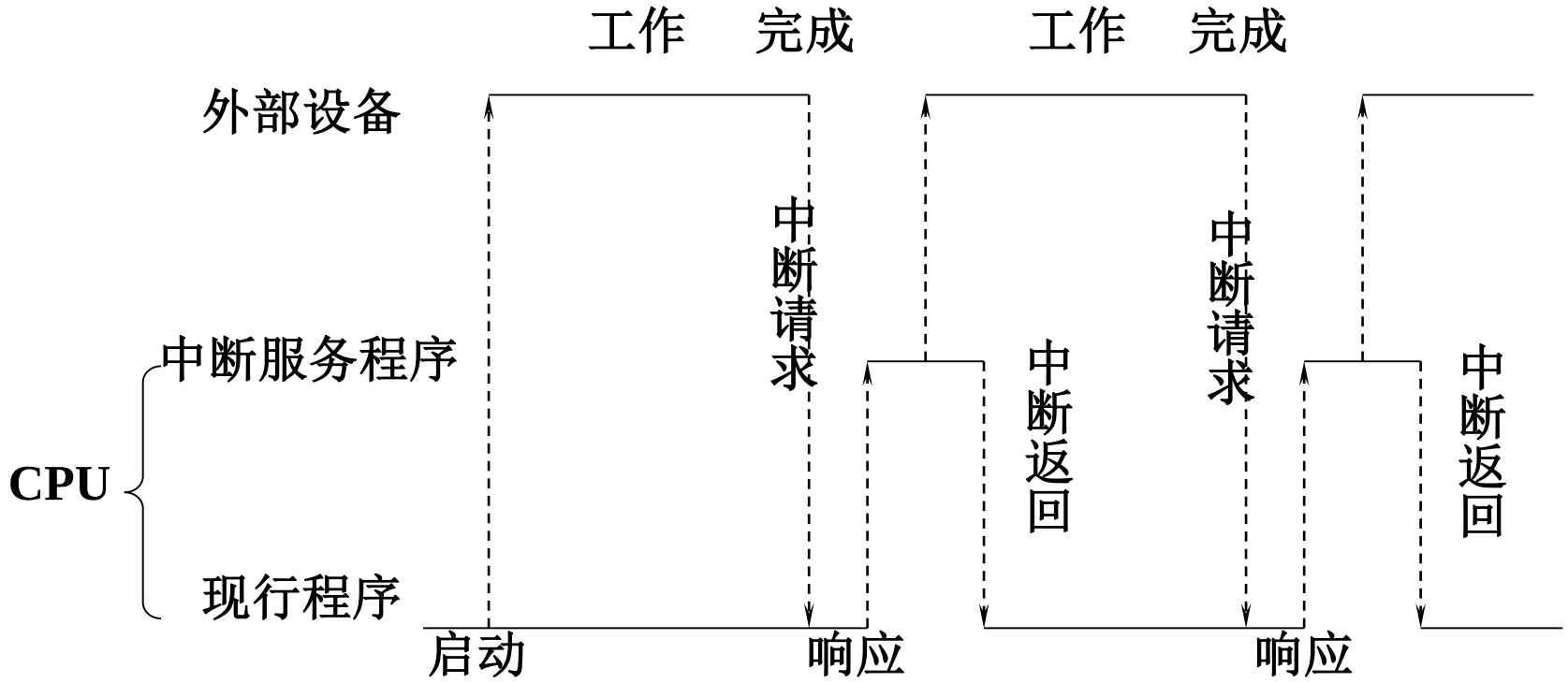
输入/输出指令
(IN AL,数据口/OUT 数据口,AL)

6.4.2 中断系统和程序中断方式

- 1. 中断的提出
  - ➢程序查询方式存在着下列明显的缺点。
    - □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。
    - □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。
    - □不能发现和处理预先无法估计的错误和异常情况。

6.4.2 中断系统和程序中断方式

- 1. 中断的提出



6.4.2 中断系统和程序中断方式

- 1. 中断的提出
  - ➢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。

6.4.2 中断系统和程序中断方式

- 2. 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。
  - 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。

6.4.2 中断系统和程序中断方式

- 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。
  - （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。

6.4.2 中断系统和程序中断方式

- 2. 中断请求信号的传送
  - （3）二维结构

    口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。
- 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。

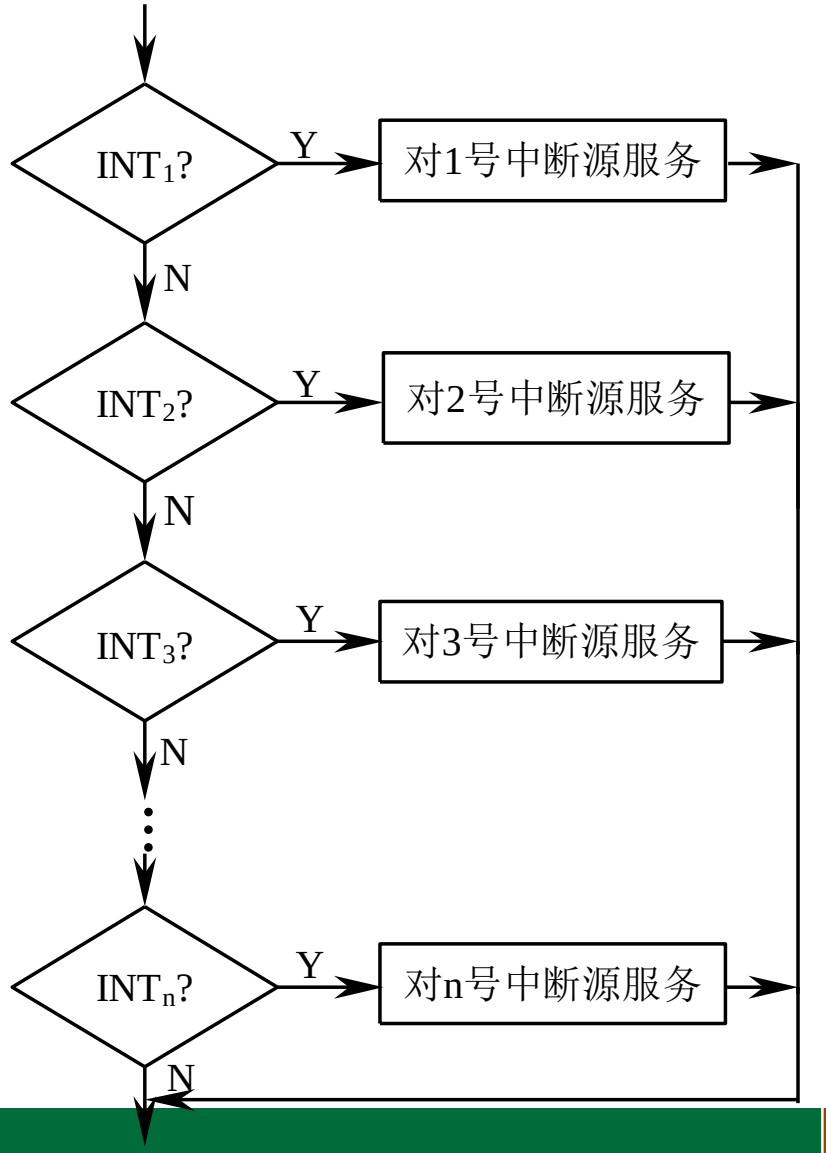
6.4.2 中断系统和程序中断方式

- 3. 中断优先级与判优方法
  - 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中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。
  - 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。

6.4.2 中断系统和程序中断方式

- 3. 中断优先级与判优方法
  - （1）软件判优法

口软件判优法，就是用程序来判别优先级，这是最简单的中断判优方法



6.4.2 中断系统和程序中断方式

- 3. 中断优先级与判优方法
  - （1）软件判优法
    - 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。

6.4.2 中断系统和程序中断方式

- 3. 中断优先级与判优方法
  - （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

6.4.2 中断系统和程序中断方式

- 4. CPU响应中断的条件
  - (1) **CPU接受到中断请求信号**
    - 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）。
  - (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。

6.4.2 中断系统和程序中断方式

- 5. 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：
  - （1）保存断点

    口将原来程序的断点（即程序计数器PC的内容）保存起来。

6.4.2 中断系统和程序中断方式

- 5. 中断隐指令
  - (2) 暂不允许中断

    口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

    口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。
- 6. 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→MAR；

6.4.2 中断系统和程序中断方式

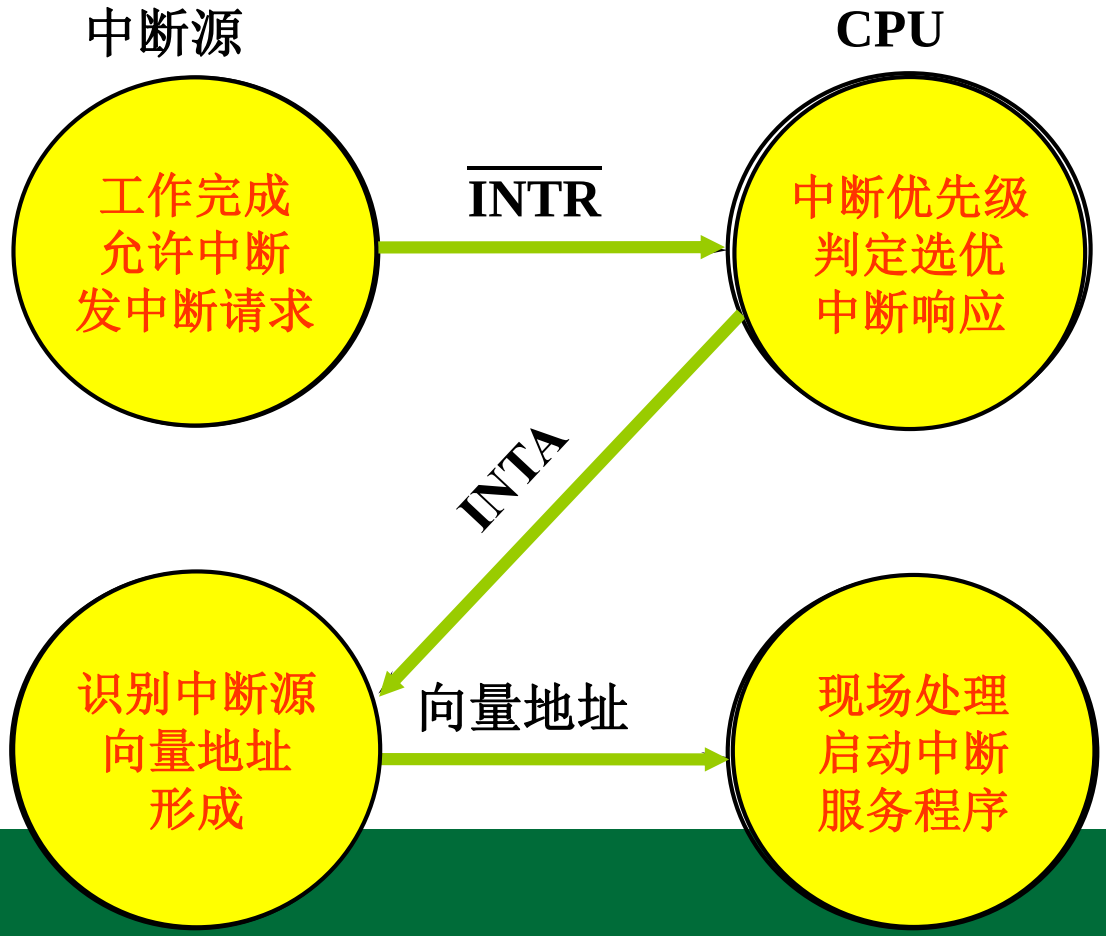
- 6. 中断周期
  - （2）将PC的内容（断点）送至MDR，记作(PC)→MDR；
  - （3）向主存发写命令，启动存储器做写操作，记作Write；
  - （4）将MDR的内容通过数据总线写入到MAR所指示的主存单元(0号）中，记作MDR→M(MAR)；
  - （5）向量地址形成部件的输出送至PC，为进入中断服务程序作准备，记作向量地址→PC；
  - （6）关中断，将中断允许触发器清0，记作0→EINT。
  - 如果断点存入堆栈，只需将上述(1)改为堆栈指针SP→MAR。

6.4.2 中断系统和程序中断方式

- 7. 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。

6.4.2 中断系统和程序中断方式

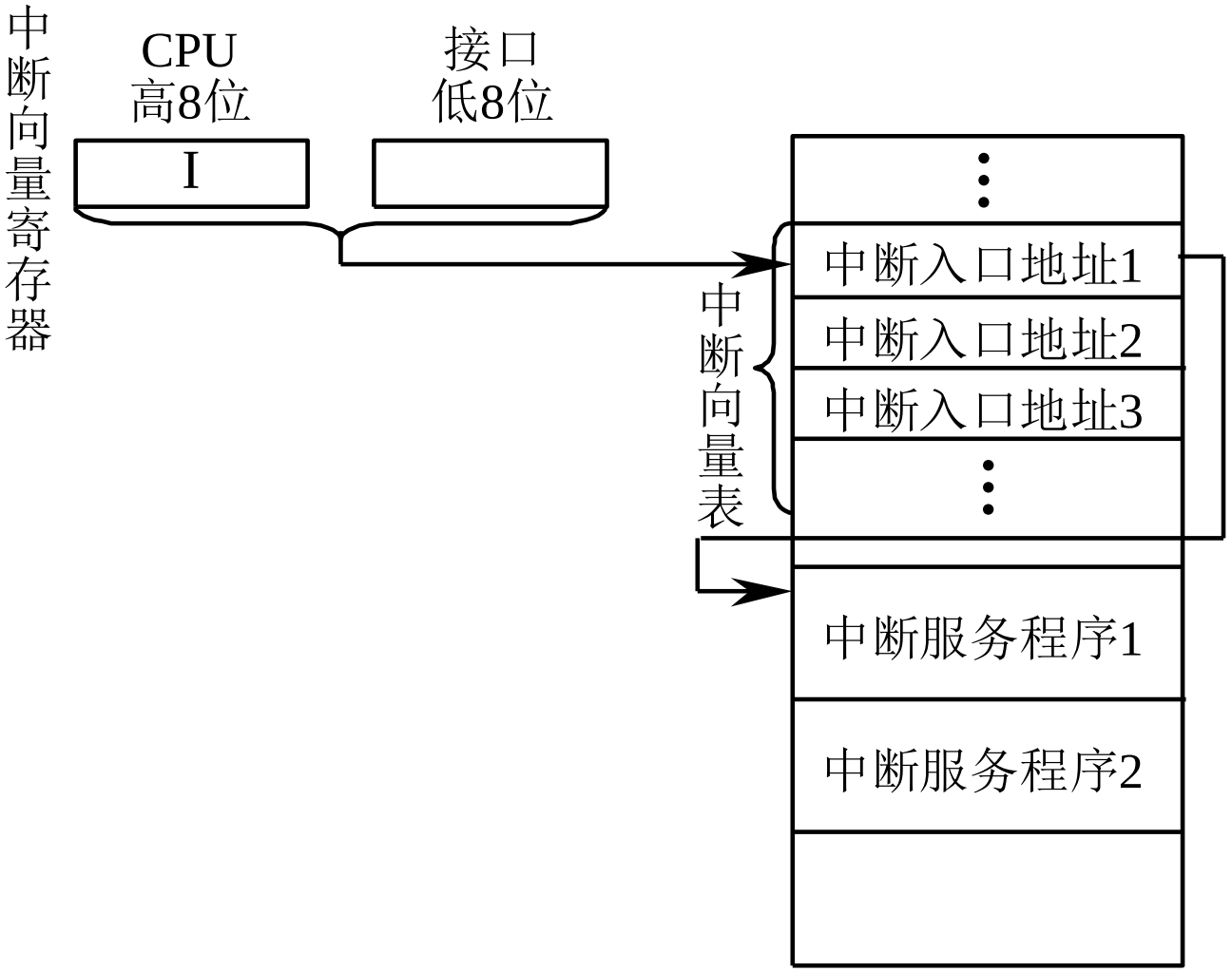
- 7. 进入中断服务程序



6.4.2 中断系统和程序中断方式

- 7. 进入中断服务程序
  - 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←8×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
  - （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。

6.4 输入输出系统控制方式



6.4.2 中断系统和程序中断方式

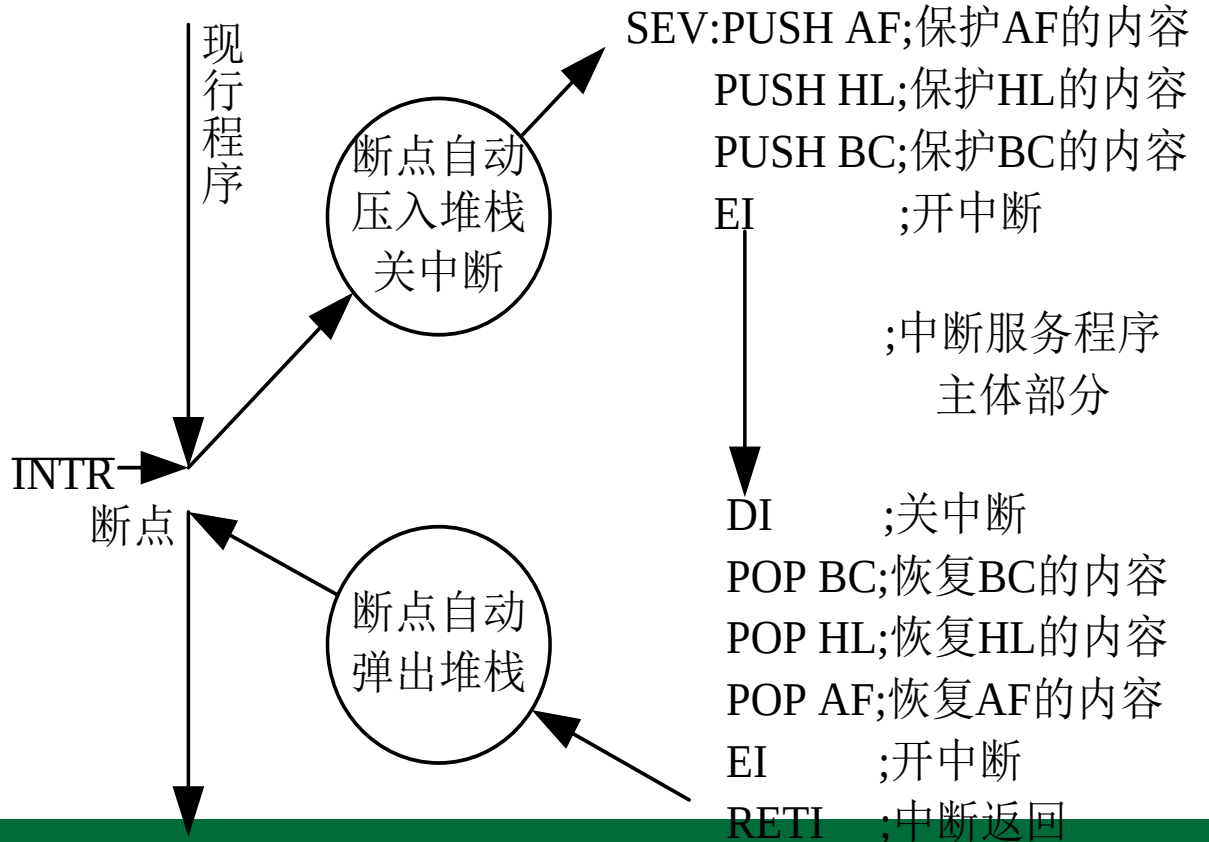
- 8. 中断现场的保护和恢复
  - 中断现场指的是发生中断时CPU的主要状态，其中最重要的是断点，另外还有一些通用寄存器的状态。之所以需要保护和恢复现场的原因是因为CPU要先后执行两个完全不同的程序（现行程序和中断服务程序），必须进行两种程序运行状态的转换。一般来说，在中断隐指令中，CPU硬件将自动保存断点，有些计算机还自动保存程序状态寄存器的内容。但是，在许多应用中，要保证中断返回后原来的程序能正确地继续运行，仅保存这一、二个寄存器的内容是不够的。

6.4.2 中断系统和程序中断方式

- 8. 中断现场的保护和恢复
  - 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。

6.4.2 中断系统和程序中断方式

- 8. 中断现场的保护和恢复



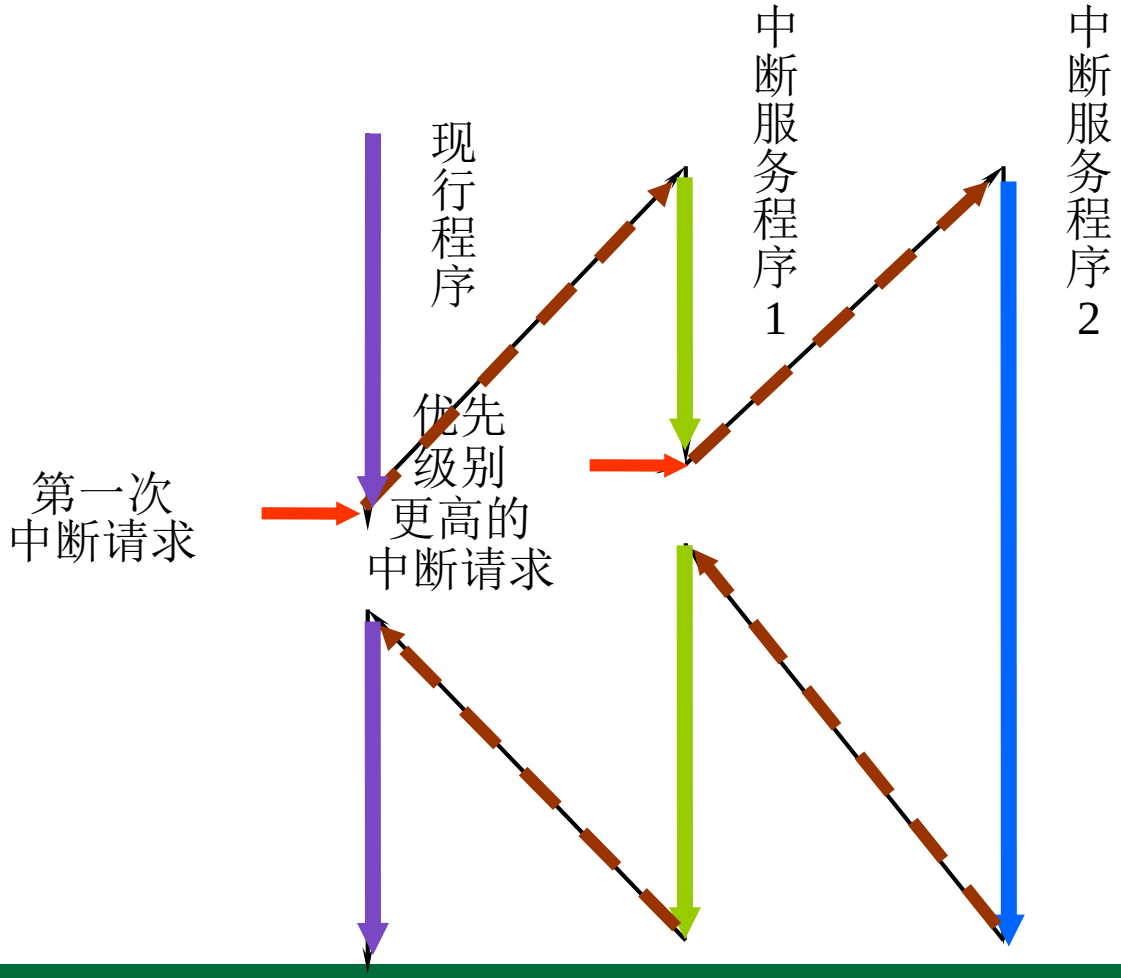
6.4.2 中断系统和程序中断方式

- 9. 中断嵌套
  - 中断嵌套的层次可以有多层， 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。

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

6.4.2 中断系统和程序中断方式

- 9. 中断嵌套



6.4.2 中断系统和程序中断方式

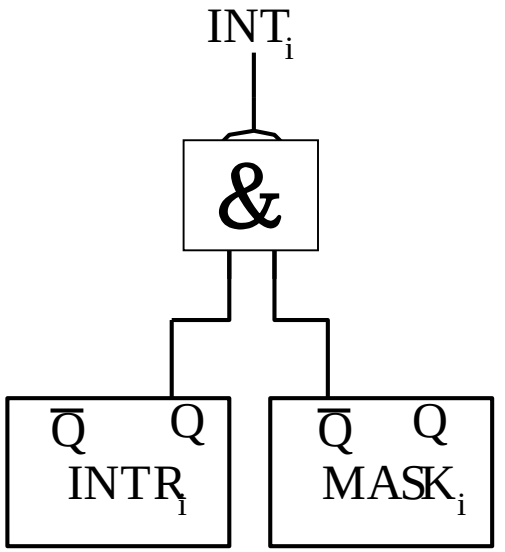
- 10. 允许和禁止中断
  - ➢允许中断还是禁止中断是用CPU中的中断允许触发器控制的，当中断允许触发器被置“1”，则允许中断，当中断允许触发器被置“0”，则禁止中断。
  - ➢允许中断即开中断，下列情况时应开中断：
    - □(1)在中断服务程序执行完毕，恢复中断现场之后；
    - □(2)在多重中断的情况下，保护中断现场之后。
  - ➢禁止中断即关中断，下列情况时应关中断：
    - □(1)当响应某一级中断请求，不再允许被其他中断请求打断时；
    - □(2)在中断服务程序的保护和恢复现场之前。

6.4.2 中断系统和程序中断方式

- 11. 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。

6.4.2 中断系统和程序中断方式

- 11. 中断屏蔽
  - 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电路之前，还要受到屏蔽触发器的控制。当MASK=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。

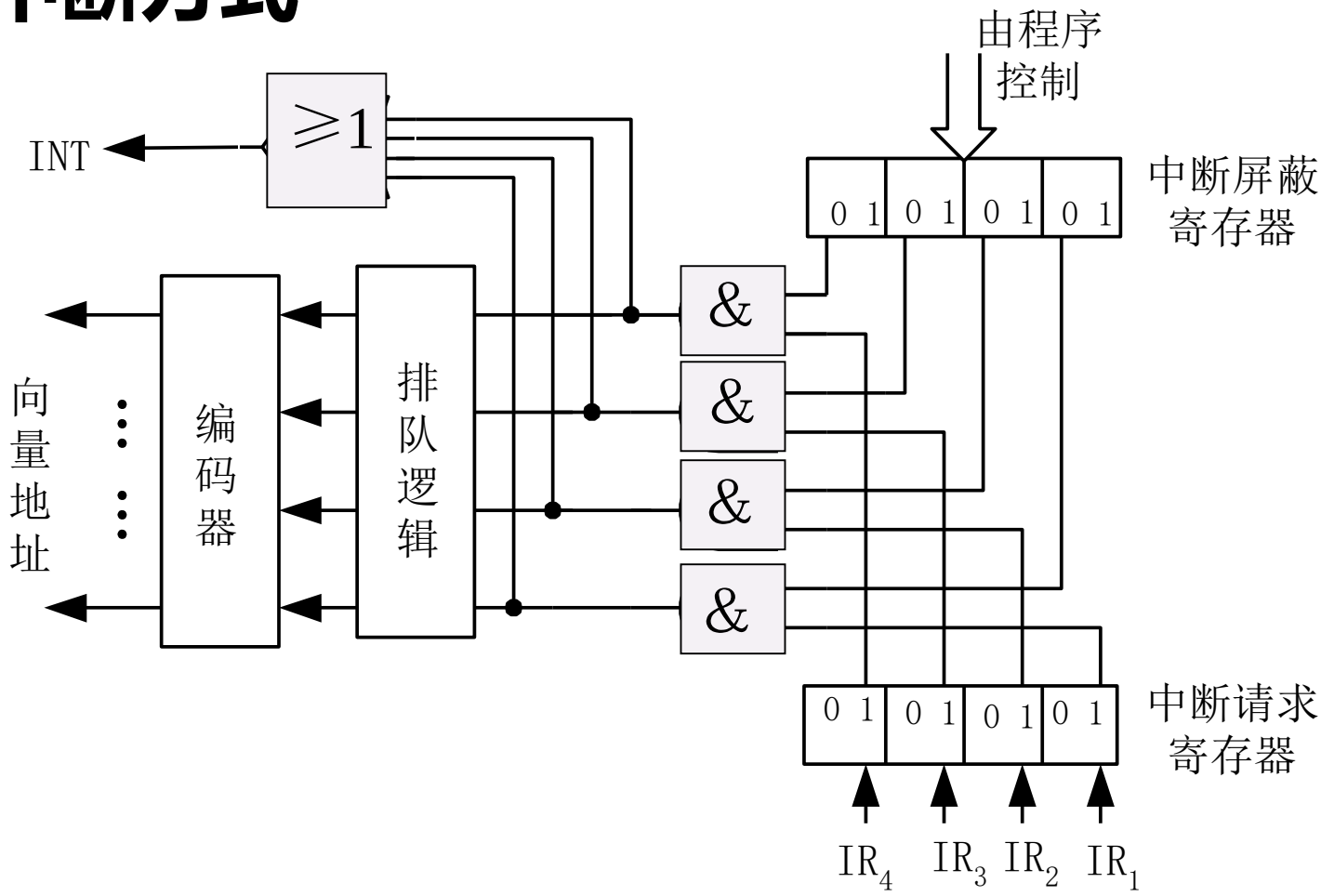


6.4.2 中断系统和程序中断方式

- 11. 中断屏蔽
  - 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。

6.4.2 中断系统和程序中断方式

- 11. 中断屏蔽



6.4.2 中断系统和程序中断方式

- 11. 中断屏蔽
  - 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。
  - 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

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

6.4.2 中断系统和程序中断方式

- 12. 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。

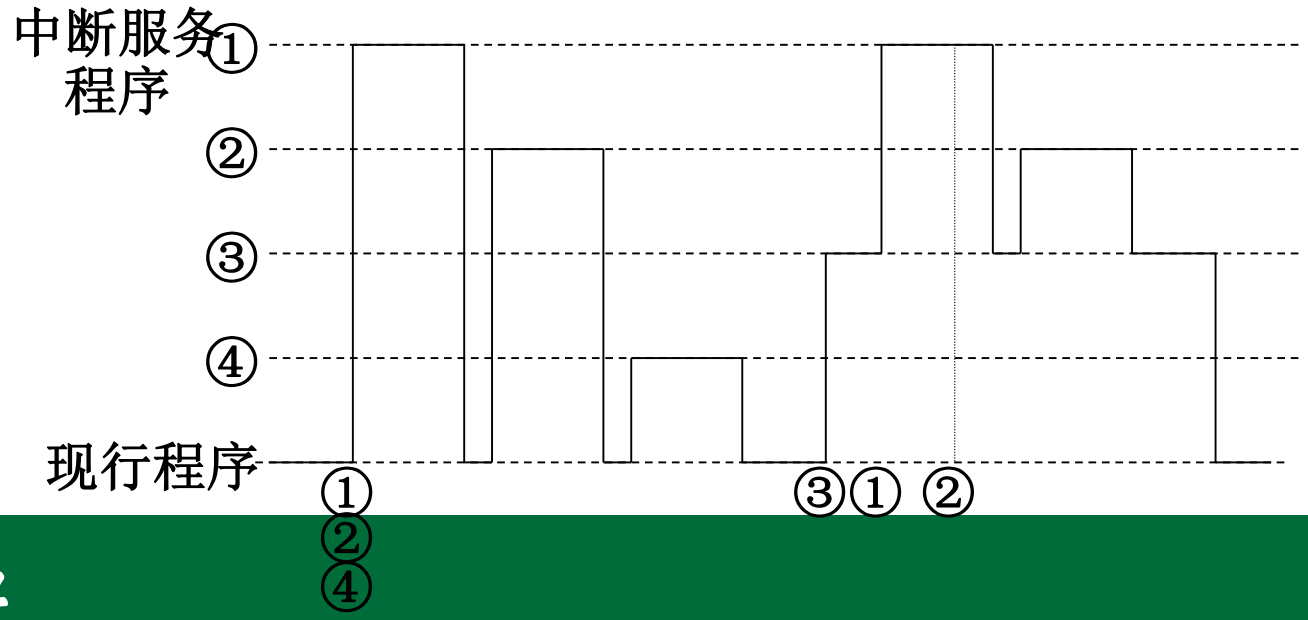
6.4.2 中断系统和程序中断方式

- 12. 中断升级
  - 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

6.4.2 中断系统和程序中断方式

- 12. 中断升级
  - 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。



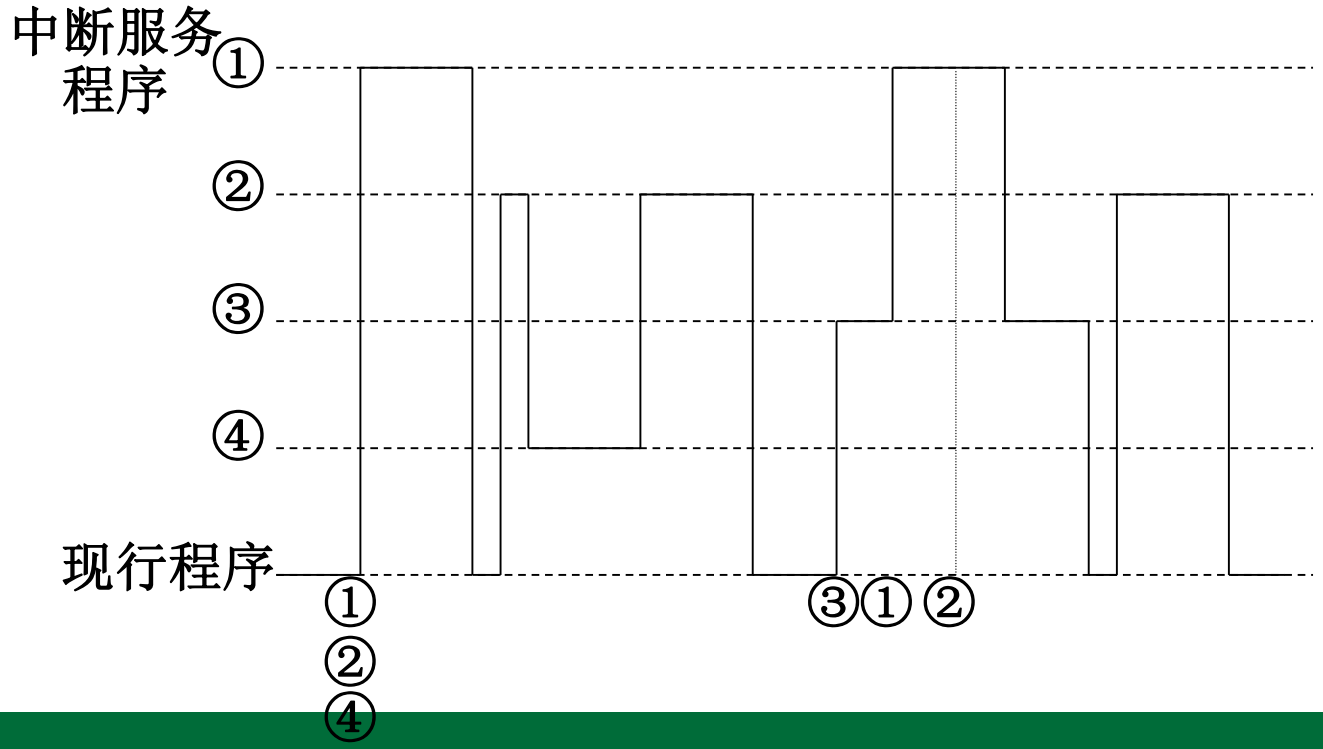
6.4.2 中断系统和程序中断方式

- 13. 中断升级
  - 在不改变中断响应次序的条件下，通过改写屏蔽码可以改变中断处理次序，例如，要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

6.4.2 中断系统和程序中断方式

- 12. 中断升级
  - ➢在同样中断请求的情况下，CPU的运动轨迹发生了变化。



6.4.2 中断系统和程序中断方式

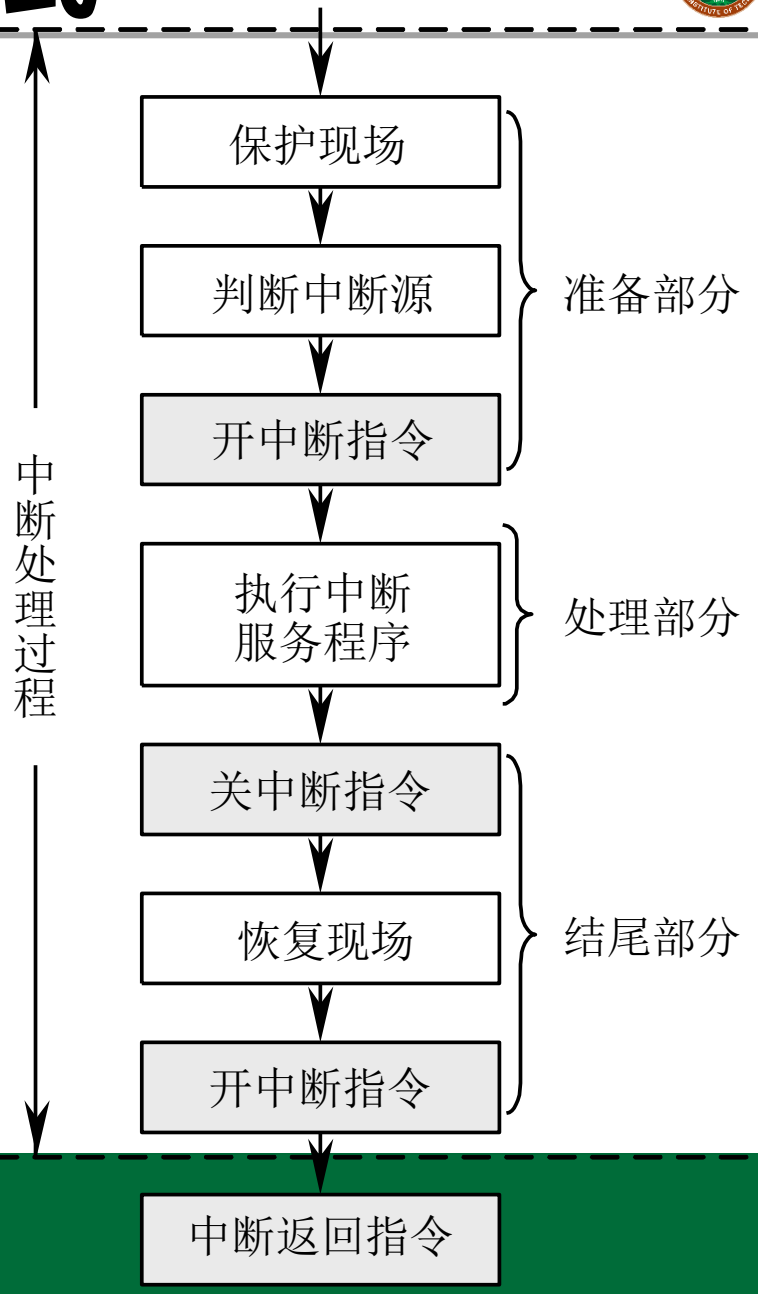
- 13. 中断全过程
  - 中断全过程是指从中断源发出中断请求开始，CPU响应这个请求，现行程序被中断，转至中断服务程序，直至中断服务程序执行完毕，CPU 再返回原来的程序继续执行的整个过程。
  - 中断全过程分为五个阶段：
  - **中断请求、中断判优、中断响应、中断处理、中断返回**。

6.4.2 中断系统和程序中断方式

- 13. 中断全过程
  - 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。

6.4.2 中断系统和程序中断方式

- 13. 中断全过程



6.4.3 DMA方式

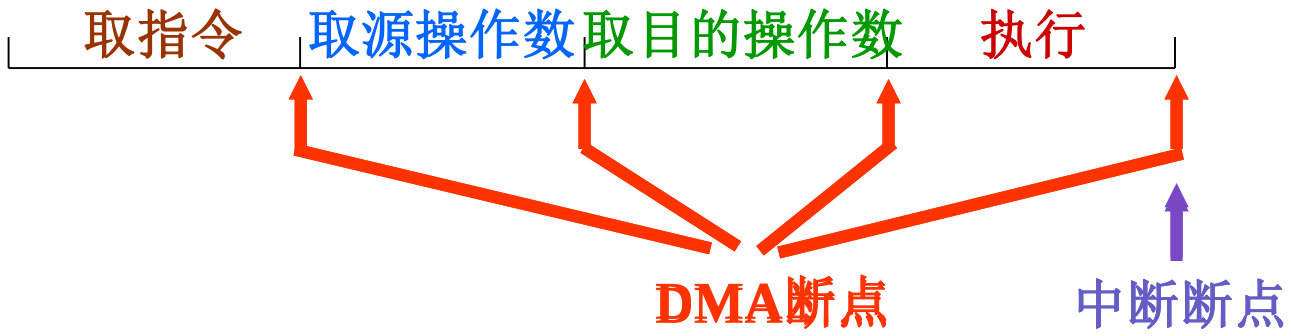
- 1. DMA方式的特点
  - 无论程序查询还是程序中断方式，主要的工作都是由CPU执行程序完成的，这需要花费时间，因此不能实现高速外设与主机的信息交换。
  - 直接存储器访问DMA方式是在外设和主存储器之间开辟一条“直接数据通道”，在不需要CPU 干预也不需要软件介入的情况下在两者之间进行的高速数据传送方式。
  - 在DMA传送方式中，对数据传送过程进行控制的硬件称为DMA控制器。当外设需要进行数据传送时，通过DMA控制器向CPU提出DMA传送请求，CPU响应之后将让出系统总线，由DMA控制器接管总线进行数据传送。

6.4.3 DMA方式

- 1. DMA方式的特点
  - DMA方式具有下列特点：
    - 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。
    - 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。
    - 主存中要开辟专用缓冲区，及时供给和接收外设的数据。
    - DMA传送速度快，CPU和外设并行工作，提高了系统的效率。
    - 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。

6.4.3 DMA方式

- 2. DMA方式和中断的区别
  - ➢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。



6.4.3 DMA方式

- 2. DMA方式和中断的区别
  - ➢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。

6.4.3 DMA方式

- 3. DMA接口（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。

6.4.3 DMA方式

- 3. DMA接口（DMA控制器）的功能
  - 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：
    - 接受外设发出的DMA请求，并向CPU发出总线请求；
    - 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；
    - 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；
    - 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。
    - 向CPU报告DMA操作的结束。

6.4.3 DMA方式

- 4. 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。
  - （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。

6.4.3 DMA方式

- 4. DMA控制器的基本组成
  - (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。
  - (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。

6.4.3 DMA方式

- 4. DMA控制器的基本组成
  - （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。

6.4.3 DMA方式

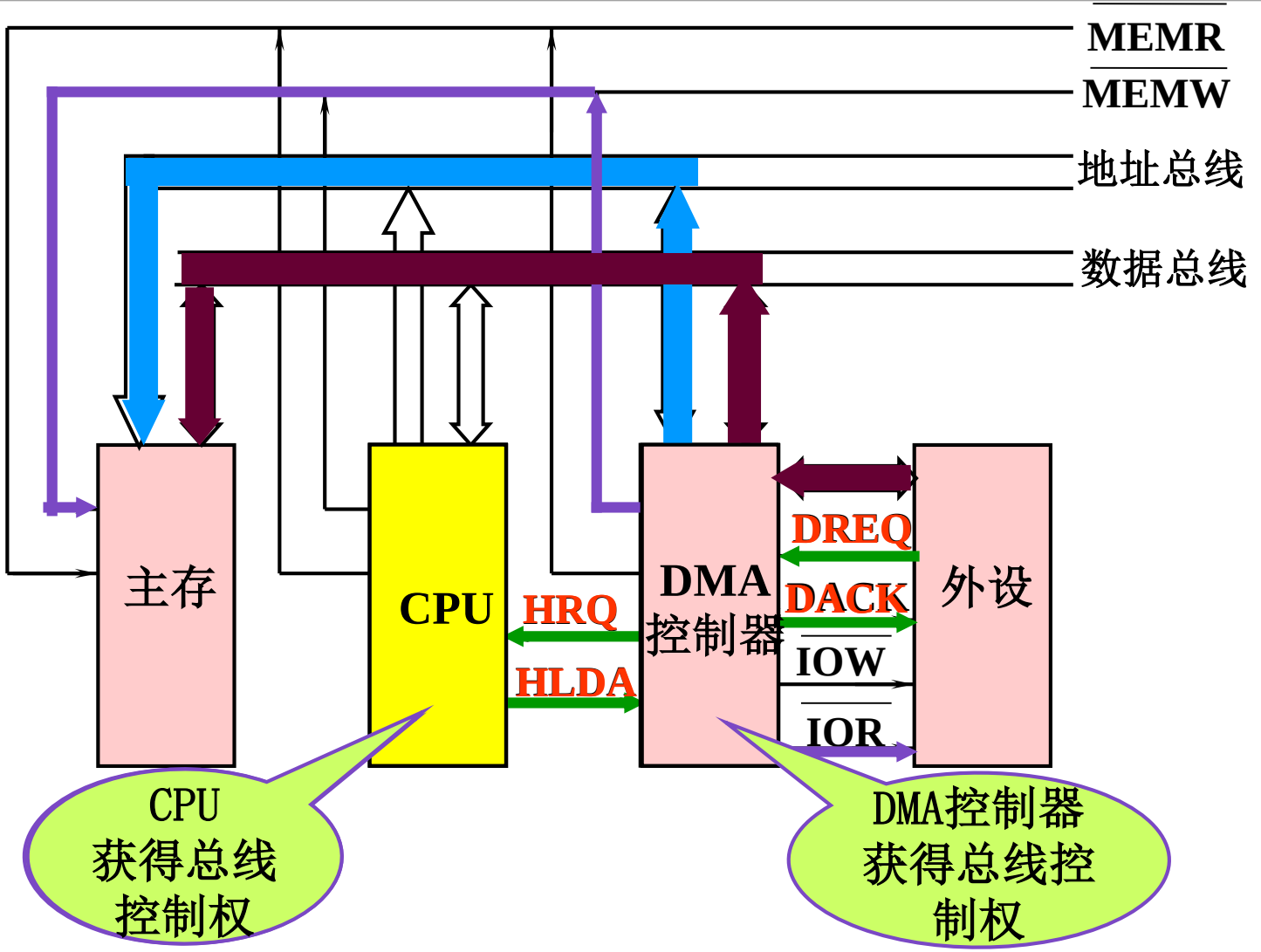
- 5. DMA控制器的引出线
  - （1）地址总线
    - 在DMA方式下，呈输出状态，可对主存进行地址选择；在CPU方式下，呈输入状态，可对DMA控制器中的有关寄存器进行寻址。
  - （2）数据总线
    - 在DMA方式下，用它进行数据传送；在CPU方式下，可对DMA控制器的有关寄存器进行编程。
  - （3）控制数据传送方式的信号线
    - 存储器读信号、存储器写信号、外设读信号、外设写信号。

6.4.3 DMA方式

- 5. DMA控制器的引出线
  - (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

6.4.3 DMA方式

- 6. DMA控制器的连接和传送



6.4.3 DMA方式

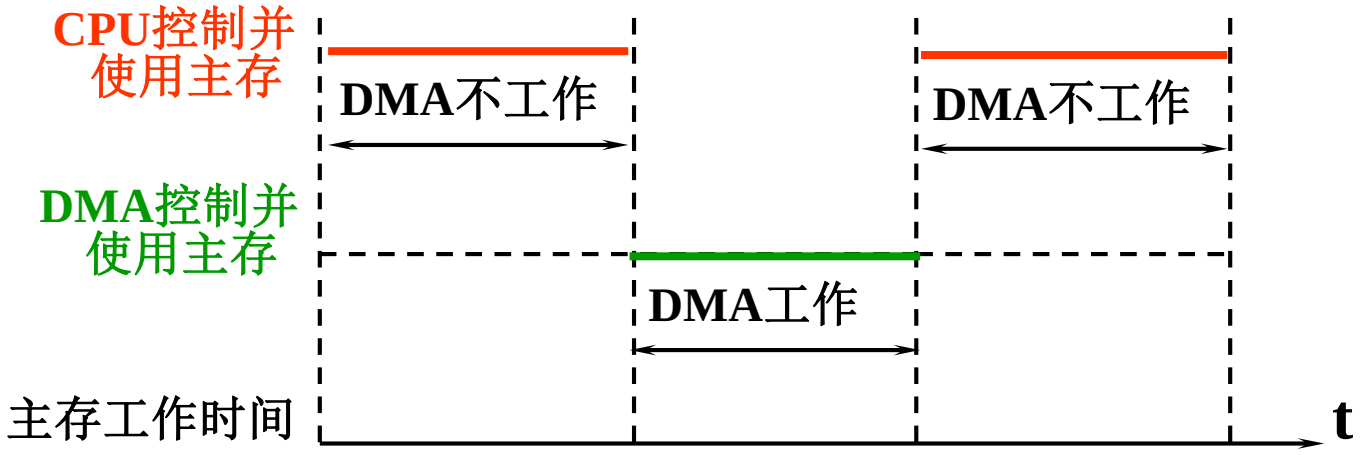
- 6. DMA控制器的连接和传送
  - (1)首先由外设向DMA控制器发出请求信号DREQ。
  - (2)DMA控制器向CPU发出总线请求信号HRQ。
  - (3)CPU向DMA控制器发出总线响应信号HLDA，此时，DMA控制器获取了总线的控制权。
  - (4)DMA控制器向外设发出DMA响应信号DACK，表示DMA控制器已控制了总线，允许外设与主存交换数据。
  - (5)DMA控制器按主存地址计数器的内容发出地址信号作为主存地址的选择，同时主存地址计数器的内容加1（或减1）。

6.4.3 DMA方式

- 6. DMA控制器的连接和传送
  - (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，将数据总线的数据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。

6.4.3 DMA方式

- 7. DMA传送方法
  - (1) CPU停止访问主存法

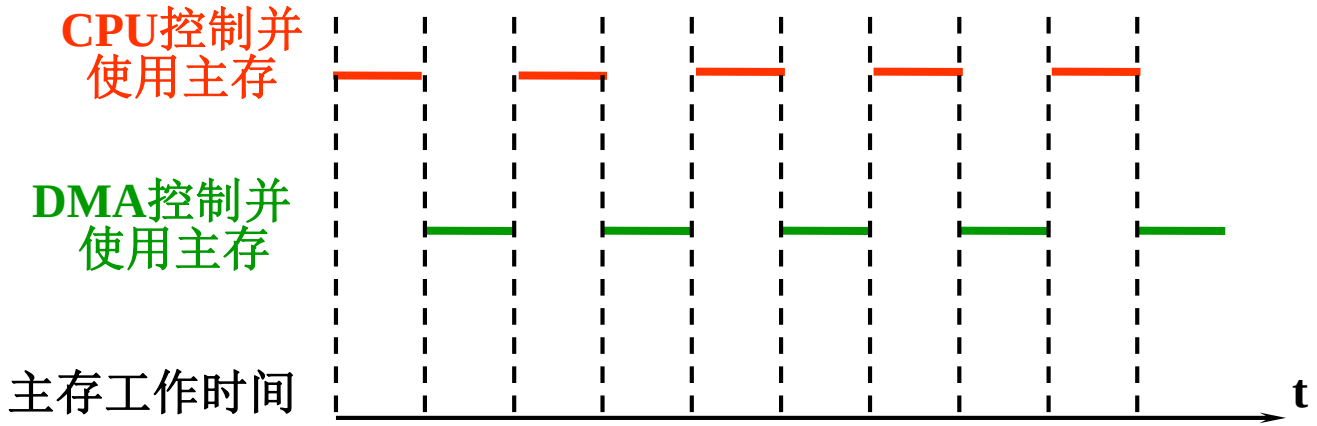


6.4.3 DMA方式

- 7. DMA传送方法
  - (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。

6.4.3 DMA方式

- 7. DMA传送方法
  - ➢(2)存储器分时法
    - 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。

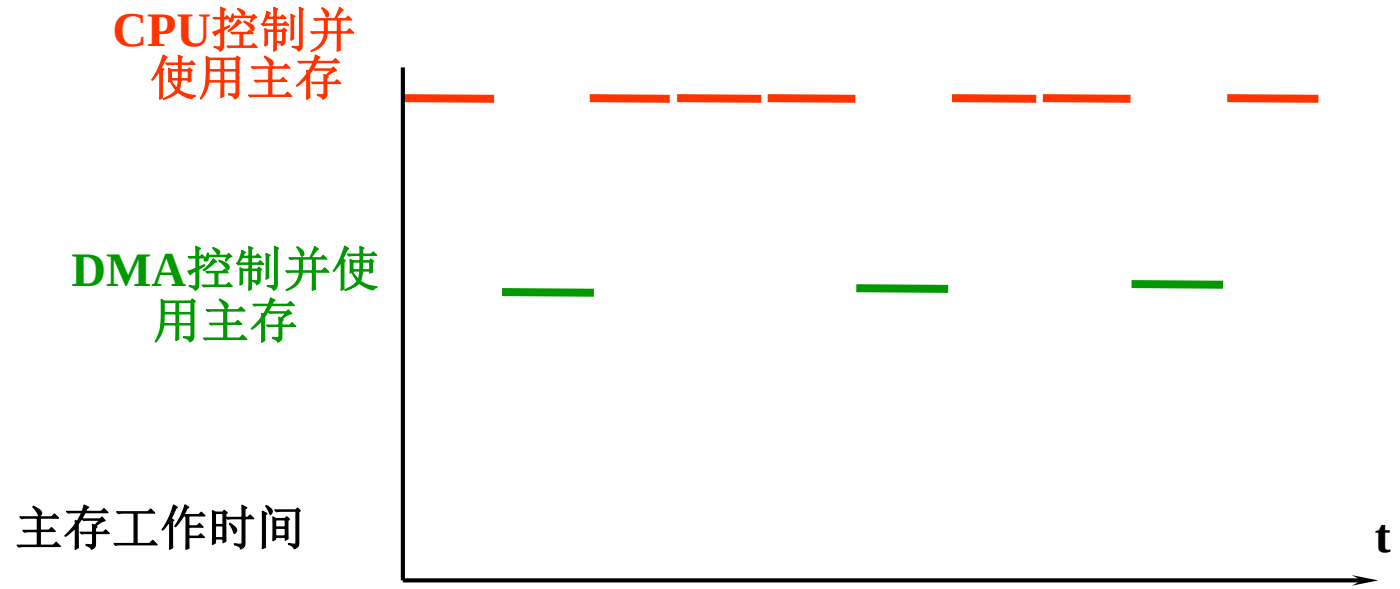


6.4.3 DMA方式

- 7. DMA传送方法
  - ➢(3)周期挪用法
    - 口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，挪用一个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。

6.4.3 DMA方式

- 7. DMA传送方法
  - (3)周期挪用法



6.4.3 DMA方式

- 8. 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。

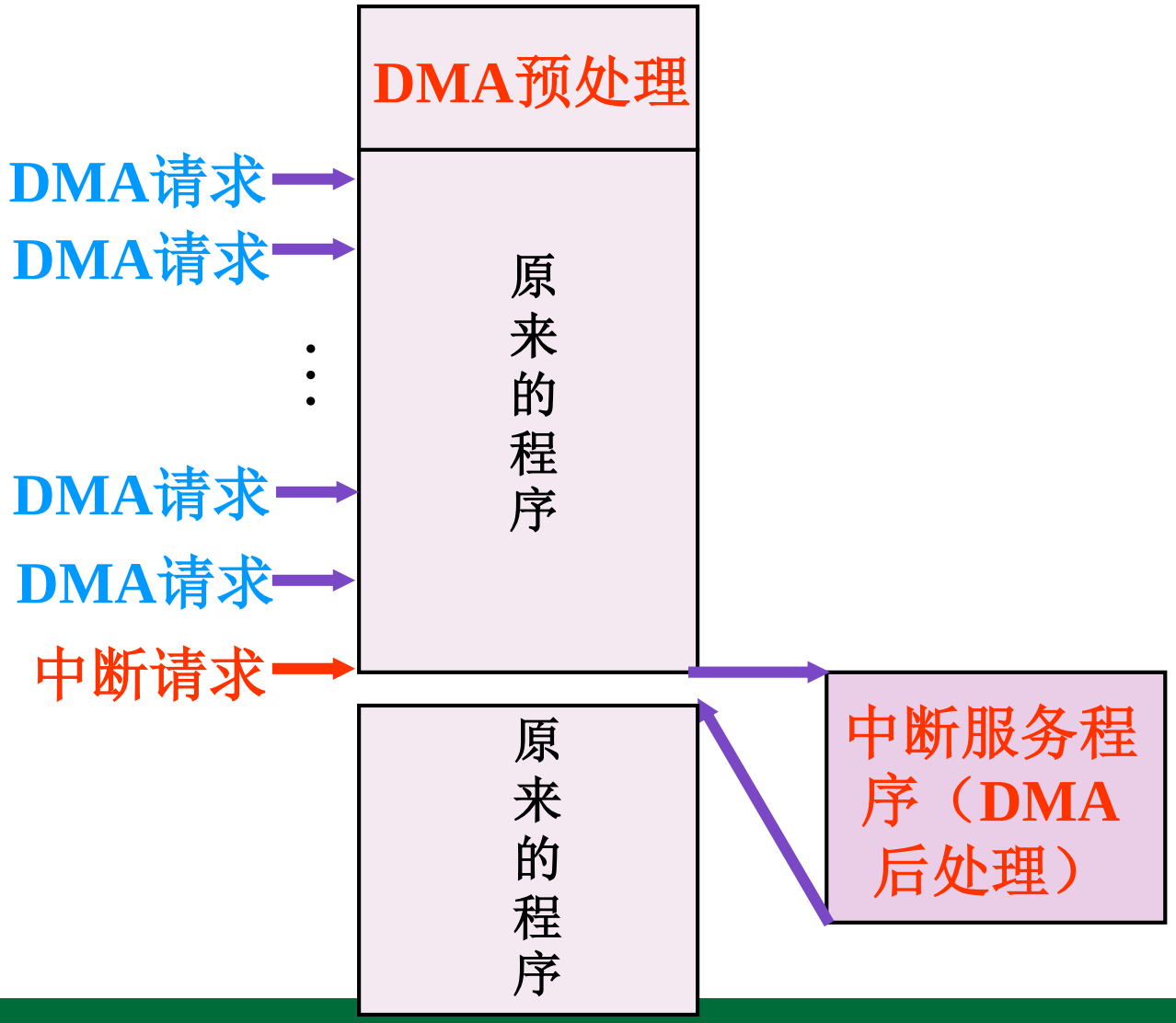
6.4.3 DMA方式

- 8. DMA传送过程
  - （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。
  - （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。

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

6.4.3 DMA方式

- 8. DMA传送过程



6.4.4 通道控制方式

- 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：

6.4.4 通道控制方式

- 1. 通道控制方式与DMA方式的区别

  口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。

  口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。

6.4.4 通道控制方式

- 1. 通道控制方式与DMA方式的区别

  口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。

  口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。

6.4.4 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：

    口接受CPU的I/O指令，按指令要求与指定的外设进行联系。

6.4.4 通道控制方式

- 2. 通道的功能
  - 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。
  - 实施主存和外设间的数据传送。
  - 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。
  - 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。

感谢聆听