Lecture 29: Input/Output System II

I/O传输方式

主　要　内　容

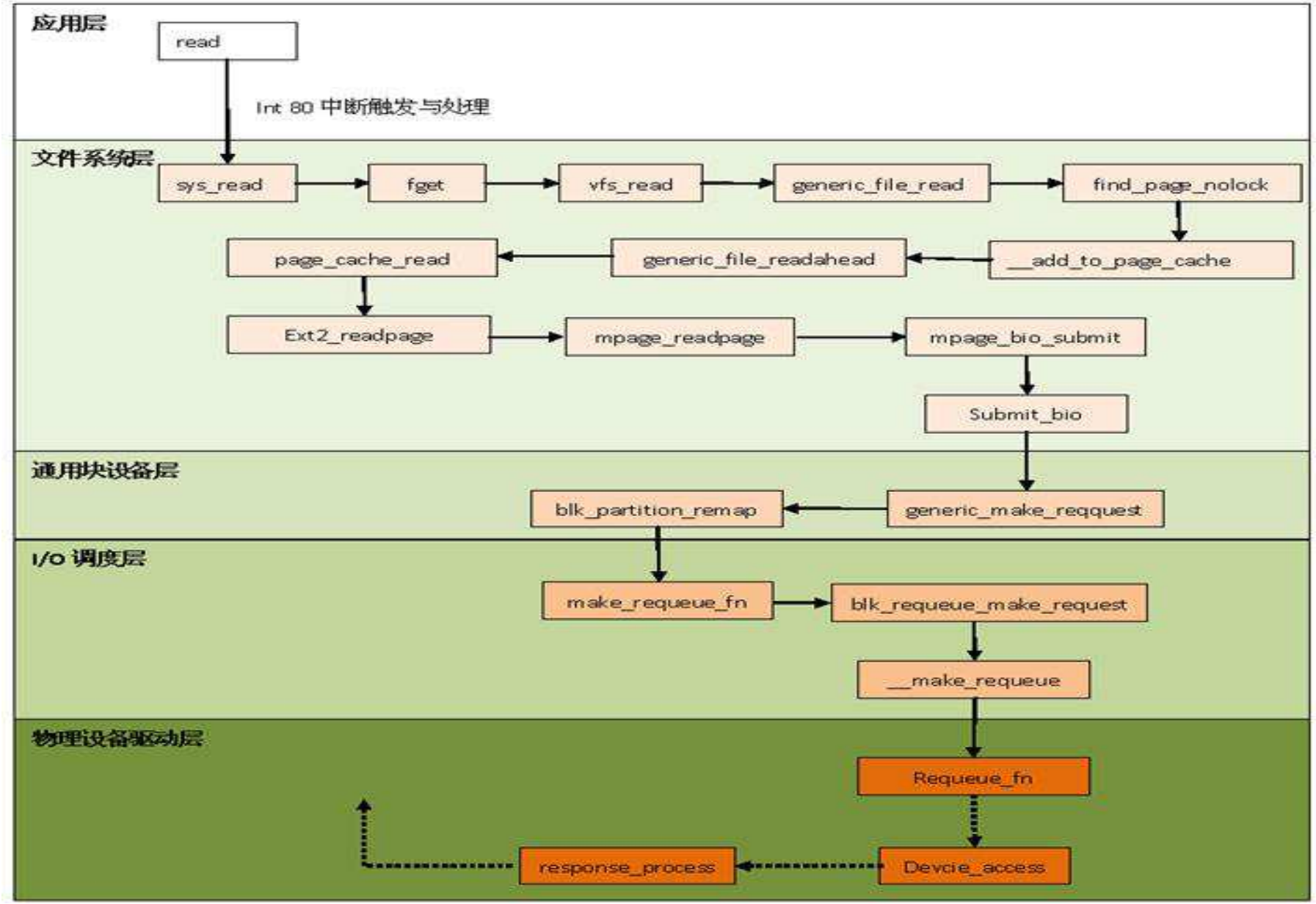
- OS在I/O系统中的职责
- I/O传输方式
  - 轮询方式（程序直接控制 / 程序查询方式）
  - 程序中断方式（中断驱动方式）
    - 中断响应的条件和中断响应过程
    - 中断处理过程
    - 中断控制器
    - 多重中断和中断屏蔽
  - 直接存储器访问方式（DMA方式）
    - DMA方式的要点
    - DMA控制器的结构
    - DMA的三种控制方式
    - DMA传输过程
  - 通道方式和I/O处理器方式简介

操作系统在I/O中扮演的角色

- OS的职责由I/O系统的三个特性决定
  - 共享性：I/O系统被处理器执行的多个程序共享，由OS统一调度管理
  - 复杂性：I/O设备控制细节复杂，不能由上层用户程序来实现，需OS提供专门驱动程序
  - 采用中断I/O方式：I/O系统通常使用外部中断请求来要求处理器执行专门的输入/出程序。中断导致向内核态转移，故必须由OS来处理
- OS在I/O中的职能
  - 保证用户程序只能访问I/O设备自己有权访问的那部分
  - 为用户程序提供设备驱动程序以屏蔽设备控制细节
  - 处理外部I/O中断，提供中断服务程序
  - 对共享的I/O资源提供合理的调度管理，使系统的吞吐率达到最佳
- 主机必须和I/O设备进行以下三类通信
  - OS必须能给I/O设备提供命令，如：磁盘寻道
  - 需要知道何时I/O设备完成操作？何时遇到什么异常问题？
  - 数据必须能在主机（主存或CPU）和I/O设备之间进行传输

这些问题就是后面这部分要讲的内容

操作系统在I/O中扮演的角色



I/O设备与主机进行数据交换的三种基本方式

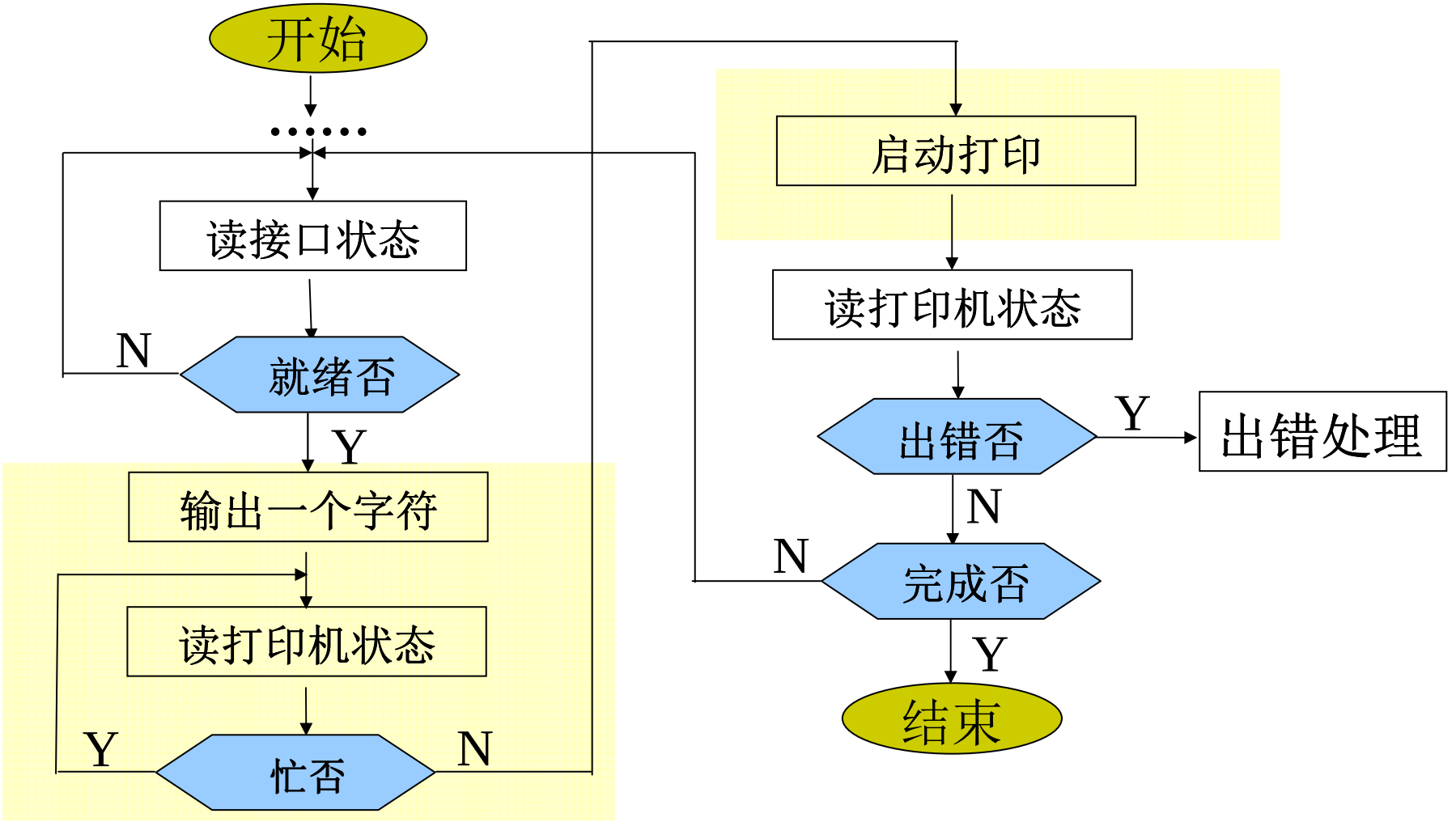
- 程序直接控制方式（最简单的I/O方式）
  - 无条件传送：对简单外设定时（同步）进行数据传送
  - 条件传送：Polling (轮询，查询): OS主动查询，也称为程序查询方式
    - I/O设备（包括I/O接口）将自己的状态放到一个状态寄存器中
    - OS阶段性地查询状态寄存器中的特定状态，以决定下一步动作
- I/O Interrupt (中断I/O): 几乎所有系统都支持的中断I/O方式
  - 若一个I/O设备需要CPU干预，它就通过中断请求通知CPU
  - CPU中止当前程序的执行，调出OS（中断处理程序）来执行
  - 处理结束后，再返回到被中止的程序继续执行
  - OS被动调出，也称为中断驱动I/O方式
- Direct Memory Access (DMA方式): 磁盘等高速外设特有的I/O方式
  - 磁盘等高速外设成批地直接和主存进行数据交换
  - 需要专门的DMA控制器控制总线，完成数据传送
  - 当外设准备好数据后，向DMA控制器发DMA请求信号，DMA控制器再向CPU发总线请求，CPU让出总线后，由DMA控制器控制总线进行传输，无需CPU干涉

程序直接控制（程序查询）方式

问题：这里“就绪”的含义是什么？

- 举例：用程序直接控制方式控制打印输出

打印控制器的数据缓冲中内容已被取走打印，现在为“空”，可以接受新的打印字符

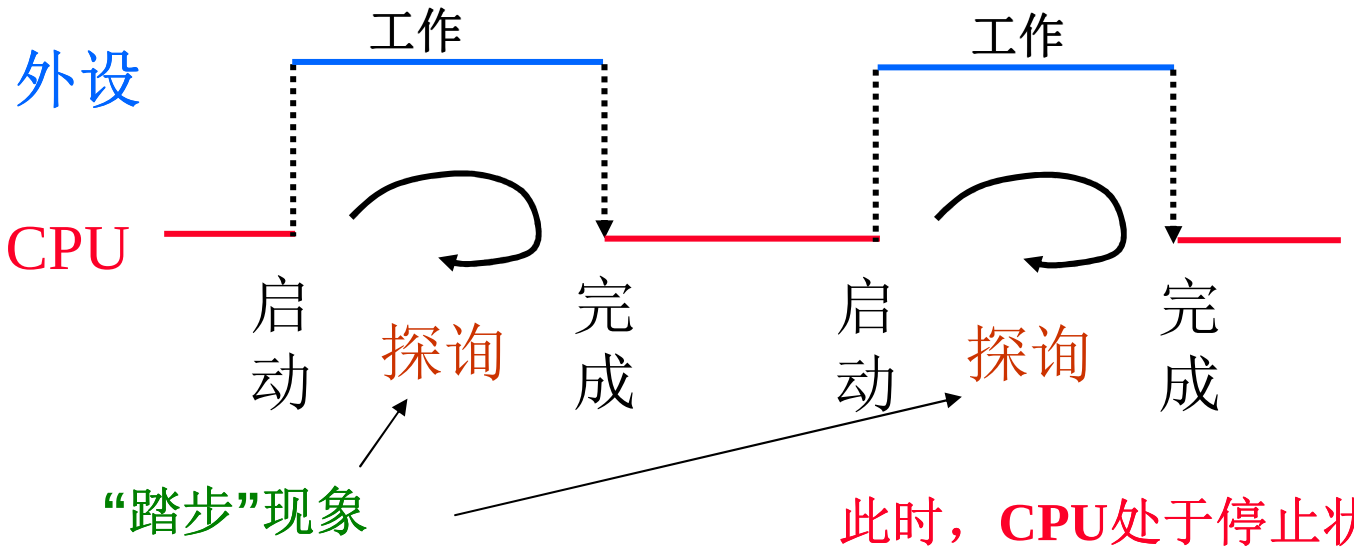


（自学）打印输出标准子程序

功能：打印AL寄存器中的字符。

```
PRINT    PROC   NEAR
         PUSH   AX          ；保留用到的寄存器
         PUSH   DX          ；保留用到的寄存器
         MOV    DX, 378H    ；输入数据锁存器口地址
         OUT    DX, AL      ；输出要打印的字符到数据锁存器
         MOV    DX, 379H    ；输入状态寄存器口地址
WAIT:    IN     AL, DX      ；读打印机状态位
         TEST   AL, 80H     ；检查忙碌位
         JE     WAIT        ；等待直到打印机不忙
         MOV    DX, 37AH    ；输入命令寄存器口地址
         MOV    AL, 0DH     ；置选通位＝1
         OUT`   DX, AL      ；使控制卡的命令锁存器中选通位置1
         MOV    AL, 0CH     ；置选通位＝0
         OUT    DX, AL      ；使控制卡的命令锁存器中选通位置0
         POP    DX
         POP    AX          ；恢复寄存器
         RET
PRINT    ENDP
```

程序控制I/O （ 查询I/O方式）



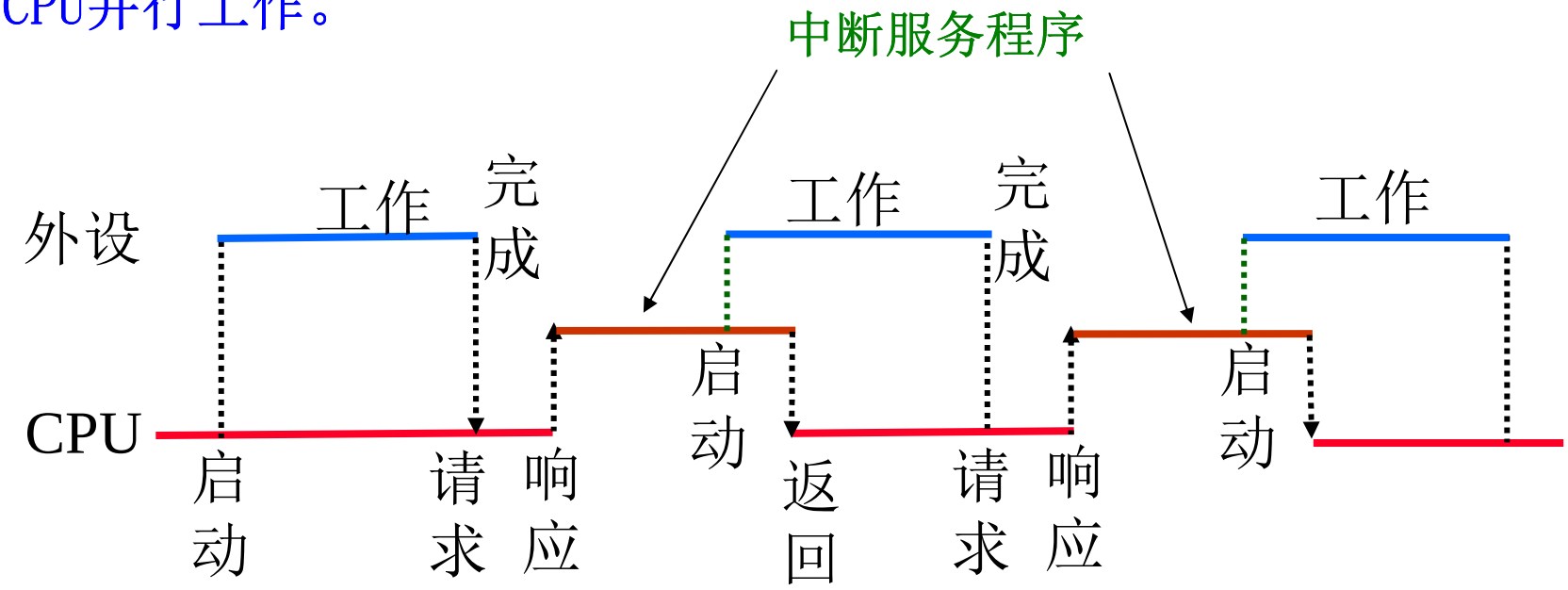
“探询”期间，可一直不断查询（独占查询），也可定时查询（需保证数据不丢失！）。

- 特点：
  - 简单、易控制、外围接口控制逻辑少；
  - CPU与外设串行工作，效率低、速度慢，适合于慢速设备
  - 查询开销极大 (CPU完全在等待“外设完成”)
- 工作方式：完全串行工作方式或部分串行，CPU用100%的时间为I/O服务！

中断I/O方式

- 基本思想：

  当外设准备好时，便向CPU发中断请求，CPU响应后，中止现行程序的执行，转入一个“中断服务程序”进行输入/出操作，实现主机和外设接口之间的数据传送，并启动外设工作。“中断服务程序”执行完后，返回原被中止的程序断点处继续执行。此时，外设和CPU并行工作。



复习：处理器中的异常处理机制

异常发生时，处理器必须做以下基本处理：

① 保护断点和程序状态：

将返回原程序执行的断点和程序状态保存到堆栈或特殊寄存器中

PC=>堆栈 或 EPC

PSWR=>堆栈 或 EPSWR

(注：PSW（Program Status Word）：程序状态字

PSWR（PSW寄存器）：用于存放程序状态的寄存器。如，X86的FLAGS）

② 识别异常事件

有两种不同的方式：软件识别和硬件识别（向量中断方式）

（1）软件识别（MIPS采用）：设置一个异常状态寄存器（**MIPS中为Cause寄存器**），用于记录异常原因。操作系统使用一个统一的异常处理程序（**MIPS的入口为0x8000 0180**），该程序按优先级顺序查询异常状态寄存器，识别出异常事件。

（2）硬件识别（向量中断）（80x86采用）：用专门的硬件查询电路按优先级顺序识别异常，得到一个“中断类型号”，根据此号，到中断向量表中读取对应的中断服务程序的入口地址。

③ 切换到具体的异常处理程序执行

问题：还有一个首先要做的基本操作，是什么？

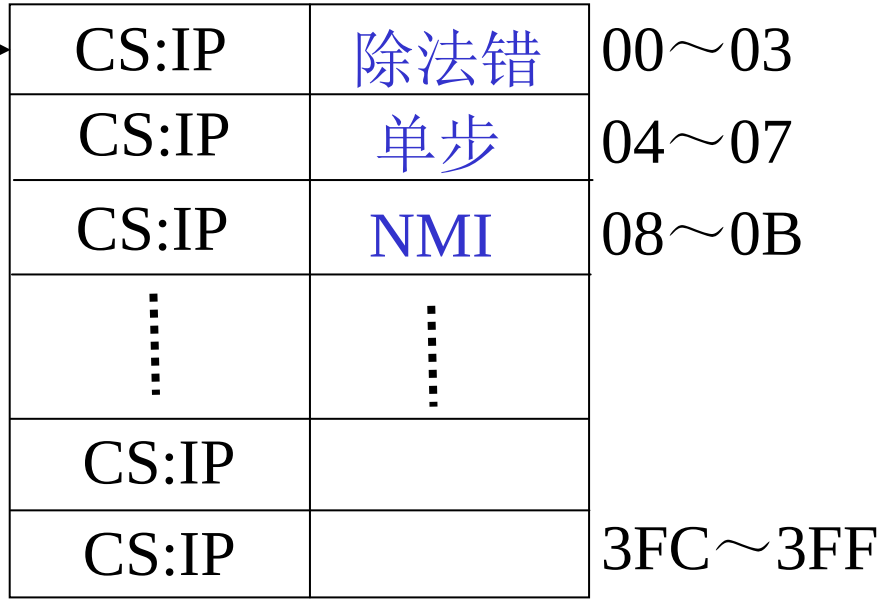
关中断！即：将中断允许标志清0。

复习：8086/8088的中断向量表

中断向量表也称中断入口地址表（或异常表），位于**0000H～03FFH**。共**256**组，每组占四个字节 **CS:IP** 。向量地址=中断类型号x4

例1：除法错的中断类型号为0，故其向量地址为：0x4=0

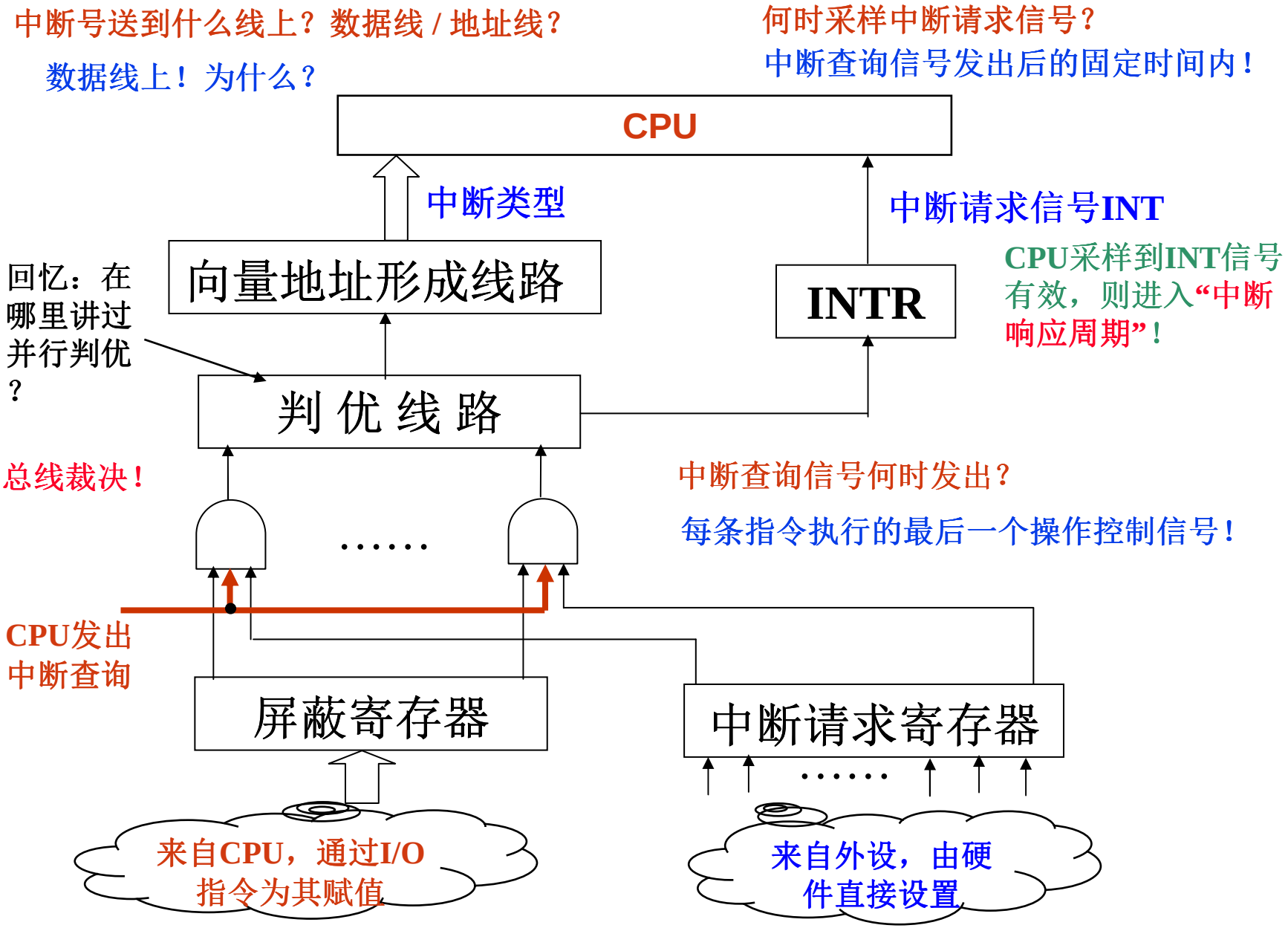
例2：NMI的中断类型号为2，故其向量地址为：2x4=8



中断向量表（异常表）中每一项是对应异常处理程序的入口地址。被称为中断向量**(Interrupt Vector)**

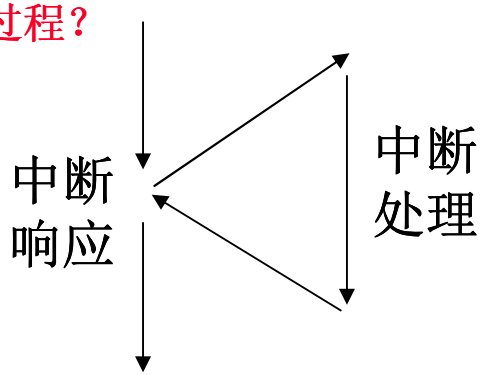
中断向量表的起始地址存放在一个异常表基址寄存器中。

中断控制器的基本结构（如：8259A）（自学）



中断驱动I/O方式

- 中断过程　回忆：在哪里讲过具体的“异常”响应过程？
  - 中断检测（硬件实现）
  - 中断响应（硬件实现）
  - 中断处理（软件实现）



- 中断响应
  - 中断响应是指主机发现外部中断请求，中止现行程序的执行，到调出中断服务程序这一过程。

(1) 中断响应的条件

① CPU处于开中断状态

② 在一条指令执行完

③ 至少要有一个未被屏蔽的中断请求

问题：中断响应的时点与异常处理的时点是否相同？为什么？

一定是在一条指令执行结束后开始查询有无中断请求，有的话立即响应，所以一定是在指令执行完时响应中断，而“异常”发生在指令执行过程中，所以不能等到指令执行完才进行异常处理。

中断I/O方式

(2)中断响应过程

执行一条隐指令，可能需完成一次总线操作，从总线上取中断类型号

具体来说，处理器做三件事：

① 关中断

0=>中断允许触发器C_{IEN}

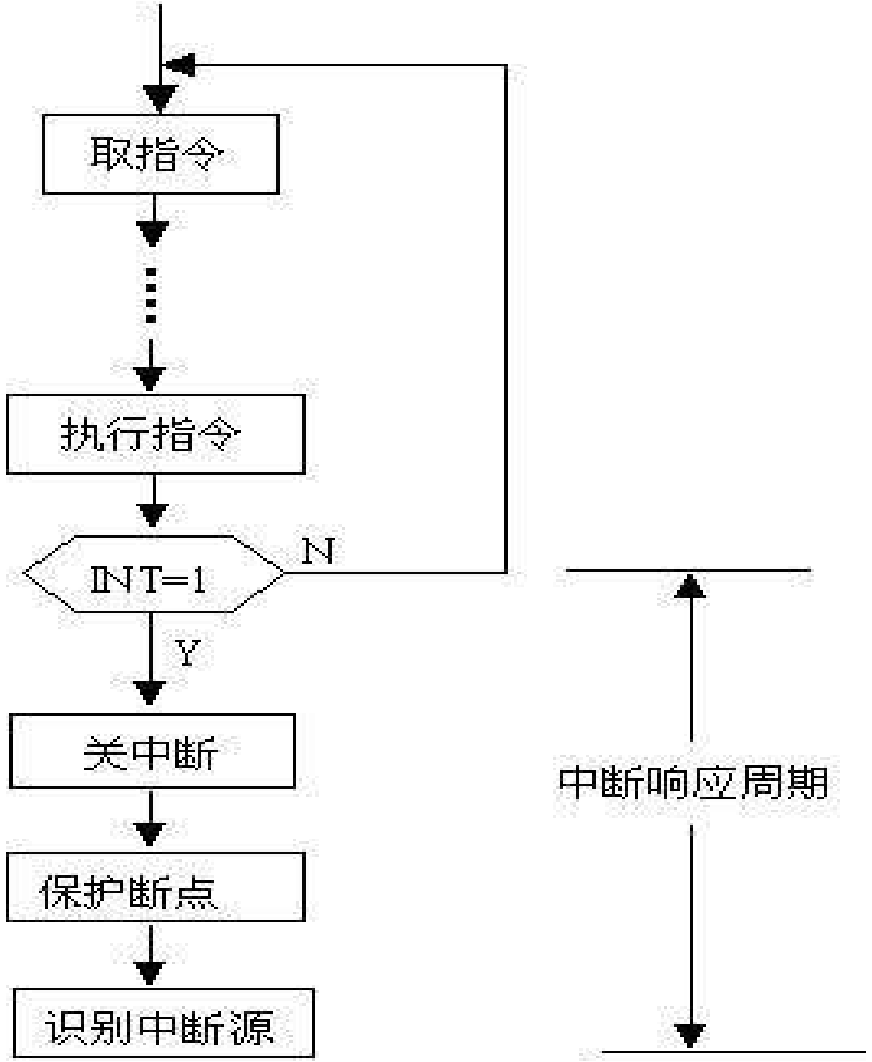
② 保护断点和程序状态

PC=>堆栈（或特殊寄存器**EPC**）

PSW=>堆栈

③ 识别中断源

取得中断服务程序首地址和初始**PSW**分别送**PC**和**PSWR**



复习：中断源的识别方法

- 软件方法（轮询）

  中断查询程序根据中断请求状态，按优先级顺序来识别

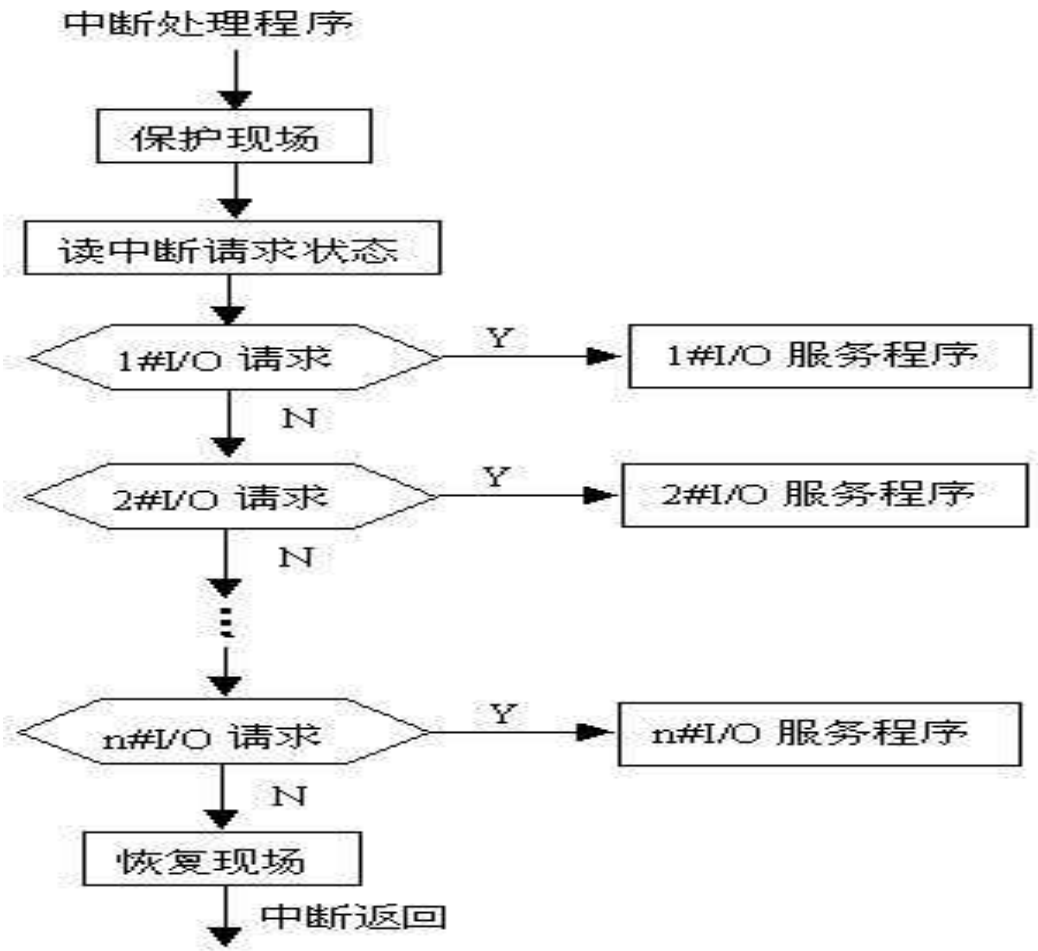
  如：**MIPS的异常/中断查询程序入口为：**

  0x8000 0180

- 硬件方法（向量中断）

  将所有中断请求状态送到一个排队电路中，根据中断优先级识别出最高优先级的中断请求

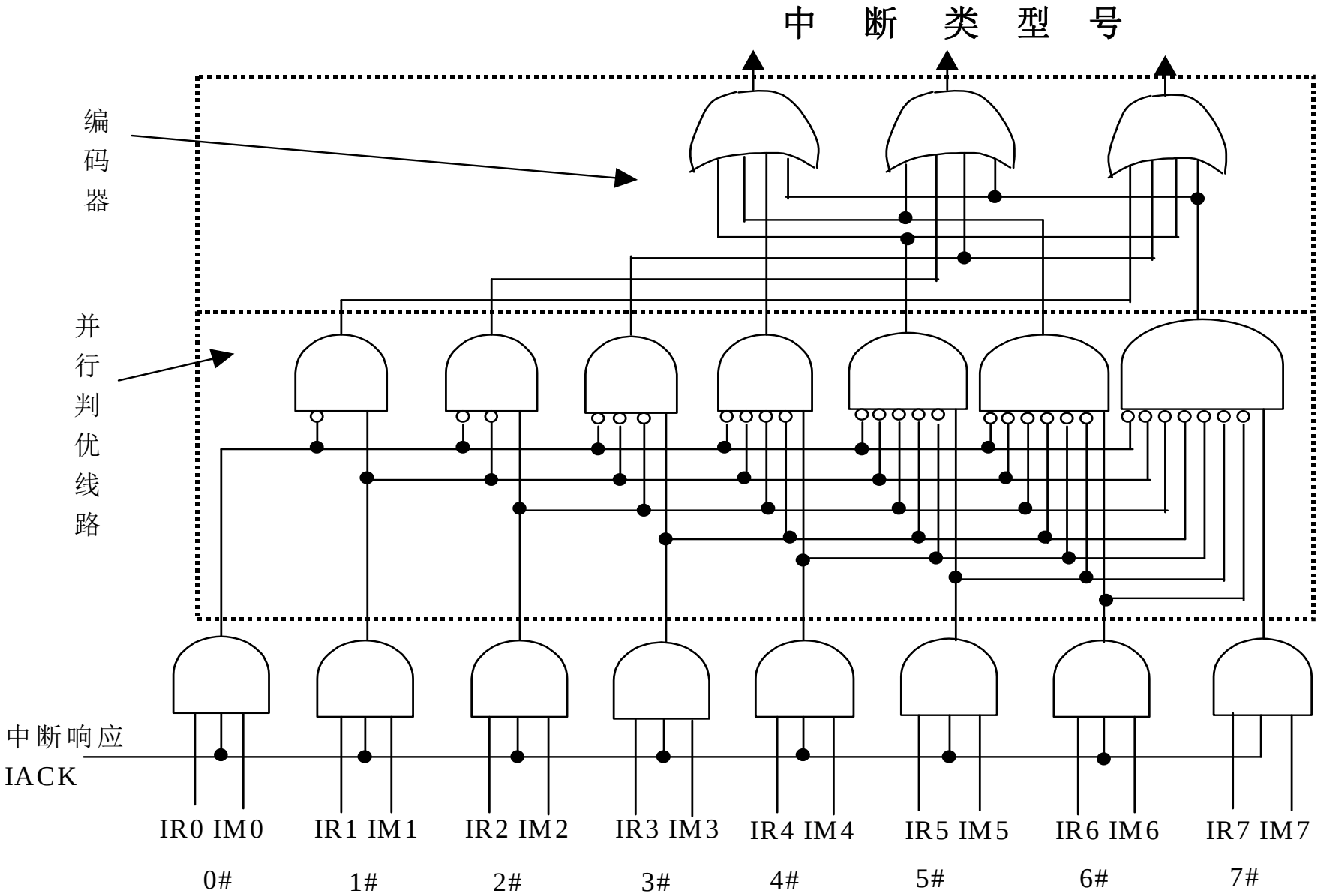
  - 链式查询（菊花链）
  - 独立请求（并行判优）



中断控制器中的“中断优先权编码器”专门用来进行中断源识别

注：内部异常和外部中断都有优先级，通常所有内部异常的优先级都比外部中断高。为什么？

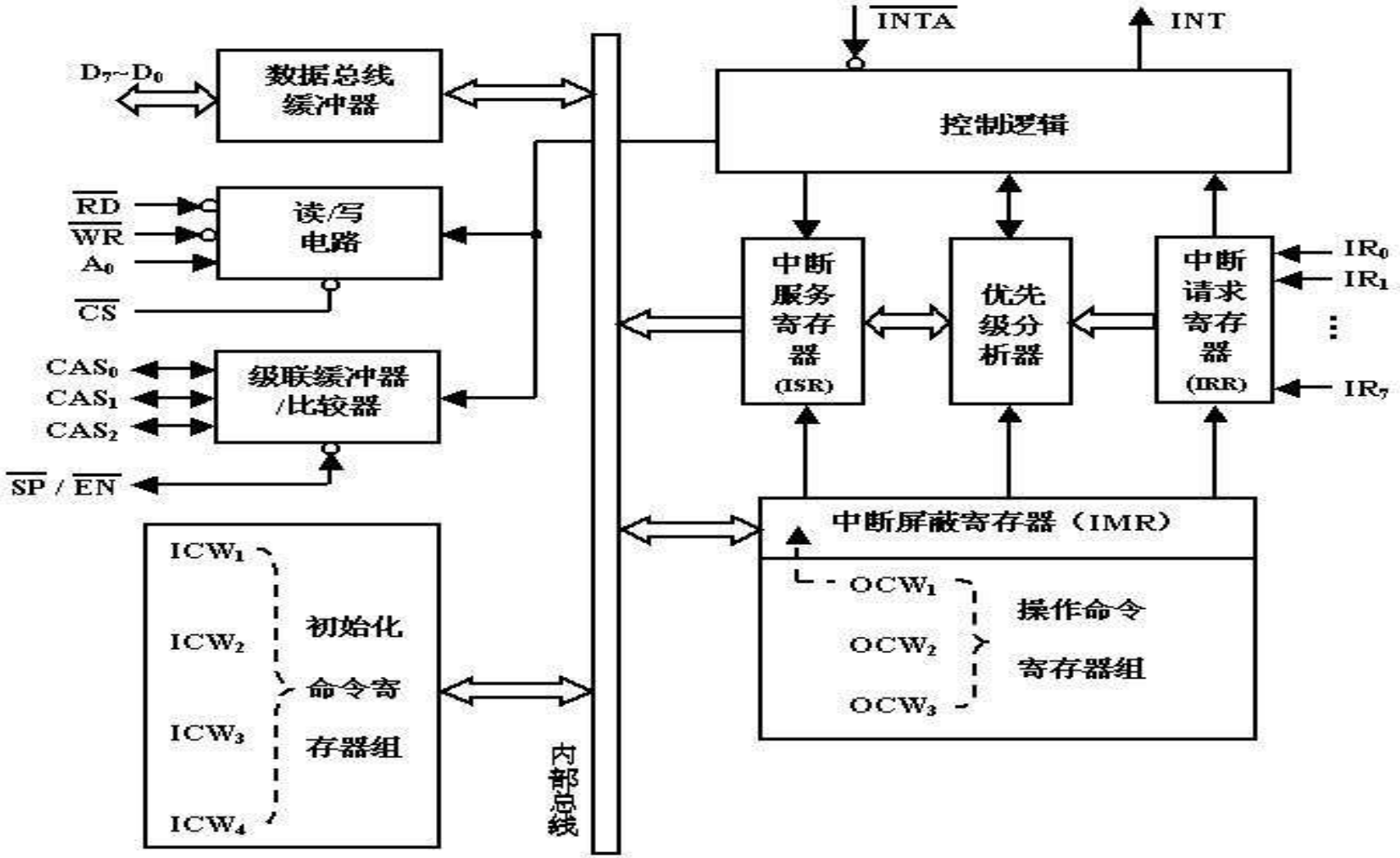
中断优先权编码器


中 断 类 型 号
编码器
并行判优线路
中断响应
IACK
IR0 IM0
IR1 IM1
IR2 IM2
IR3 IM3
IR4 IM4
IR5 IM5
IR6 IM6
IR7 IM7
0#
1#
2#
3#
4#
5#
6#
7#

（自学）中断控制器举例-8259A

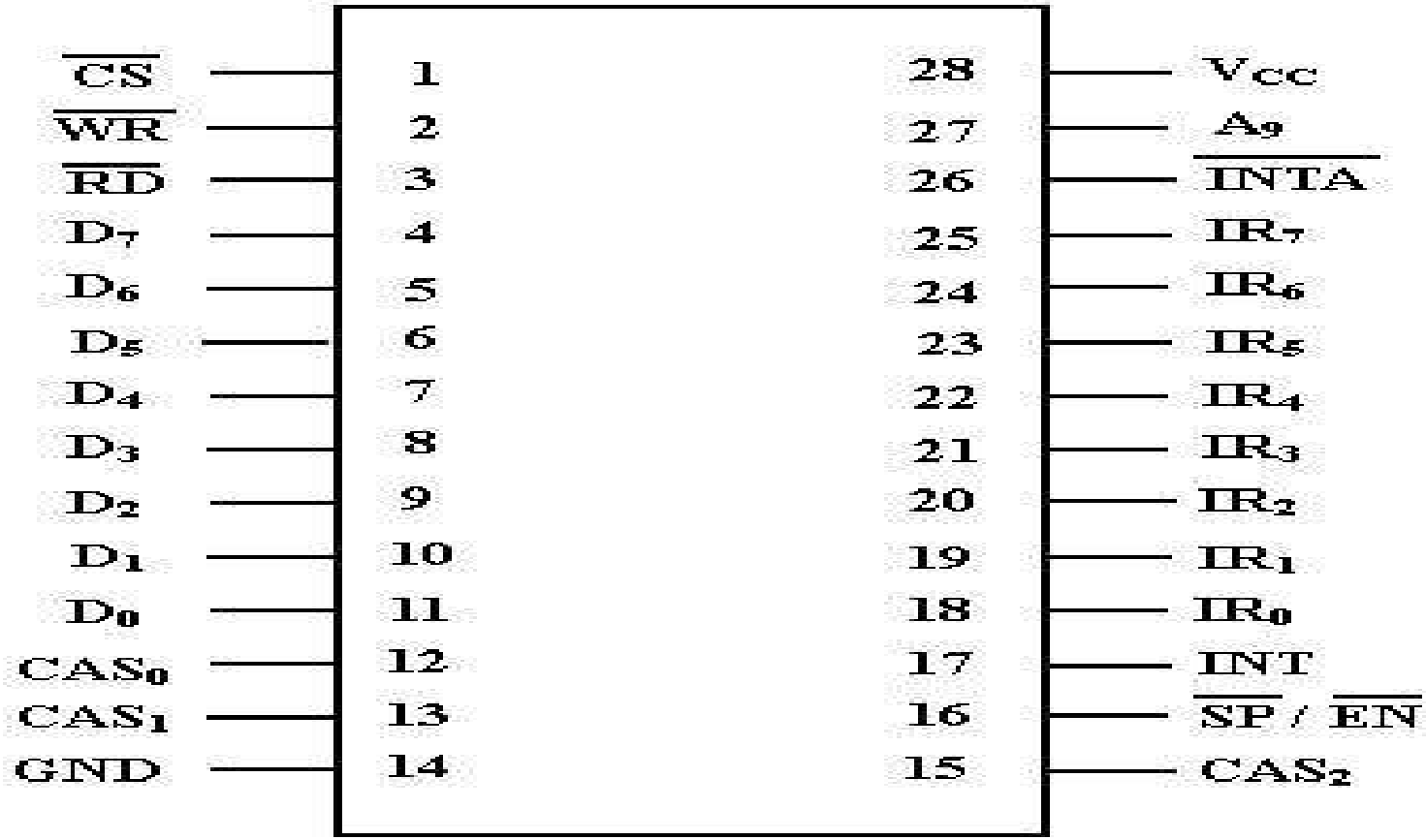
- **8259A**是可编程的中断控制器
- **8259A**的功能
  - 包含中断请求锁存、中断屏蔽、中断优先级排队、中断向量生成（中断优先权编码器）等电路
  - 既可支持程序查询式中断，又可支持向量式中断
  - 支持**8**级优先权，通过多片级联，最多可构成**64**级中断
  - 各种中断功能可通过编程设定或更改

（自学）8259A的内部结构



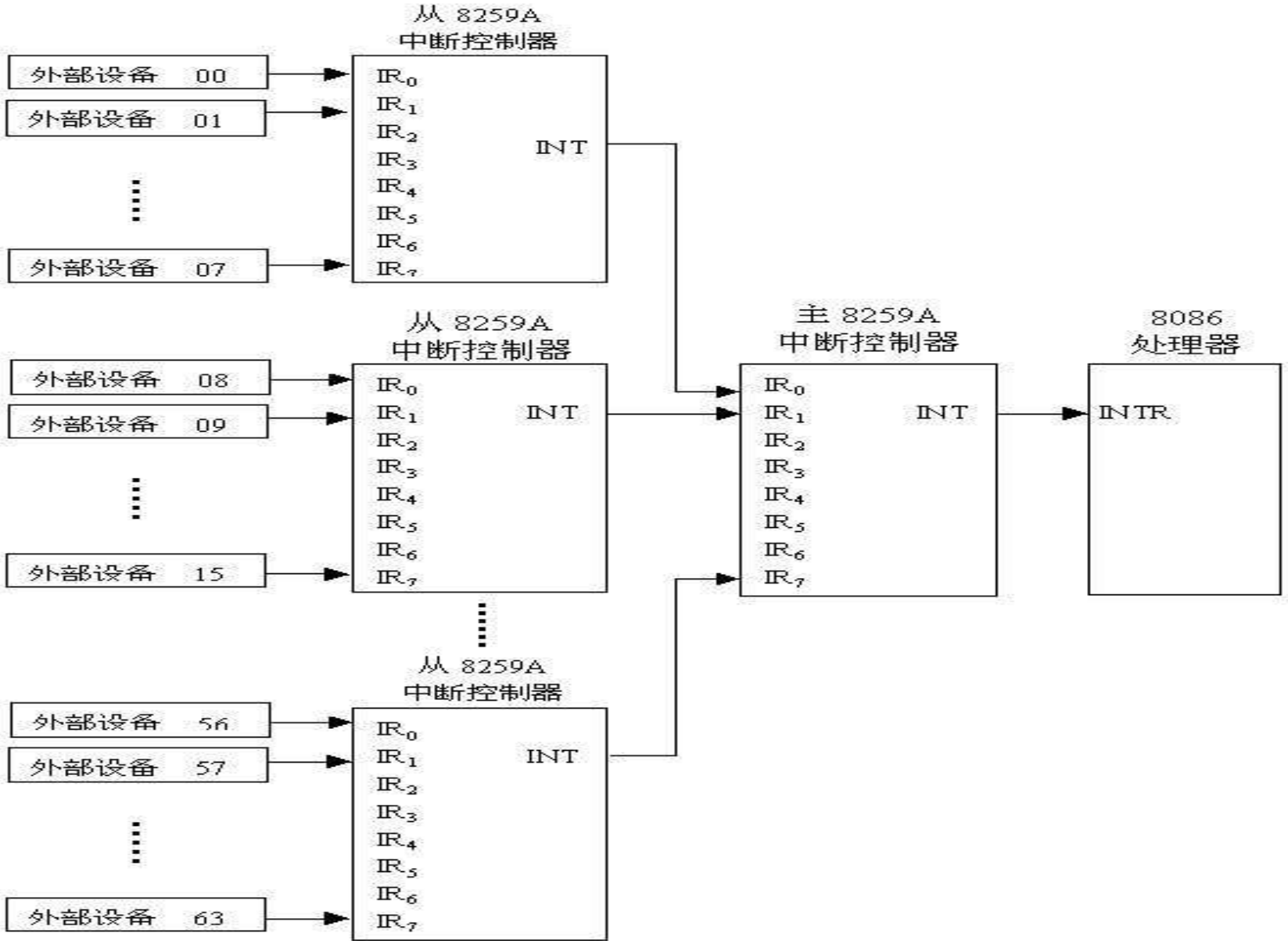
8259A 内部结构框图

（自学）8259A的引脚功能



8259A 引脚图

（自学）8259A芯片的级联



中断处理过程

中断过程：中断响应+中断处理

中断响应的结果是调出相应的中断服务程序。

- 中断处理：
  - 是指执行相应中断服务程序的过程。
  - 不同的中断源其对应的中断服务程序不同。
  - 典型的中断处理（中断服务程序）分为三个阶段：
    - 先行段（准备阶段）
      - 保护现场及旧屏蔽字
      - 查明原因（软件识别中断时）
      - 设置新屏蔽字
      - 开中断
    - 本体段（具体的中断处理阶段）
    - 结束段（恢复阶段）
      - 关中断
      - 恢复现场及旧屏蔽字
      - 清“中断请求”
      - 开中断
      - 中断返回

多重中断的概念

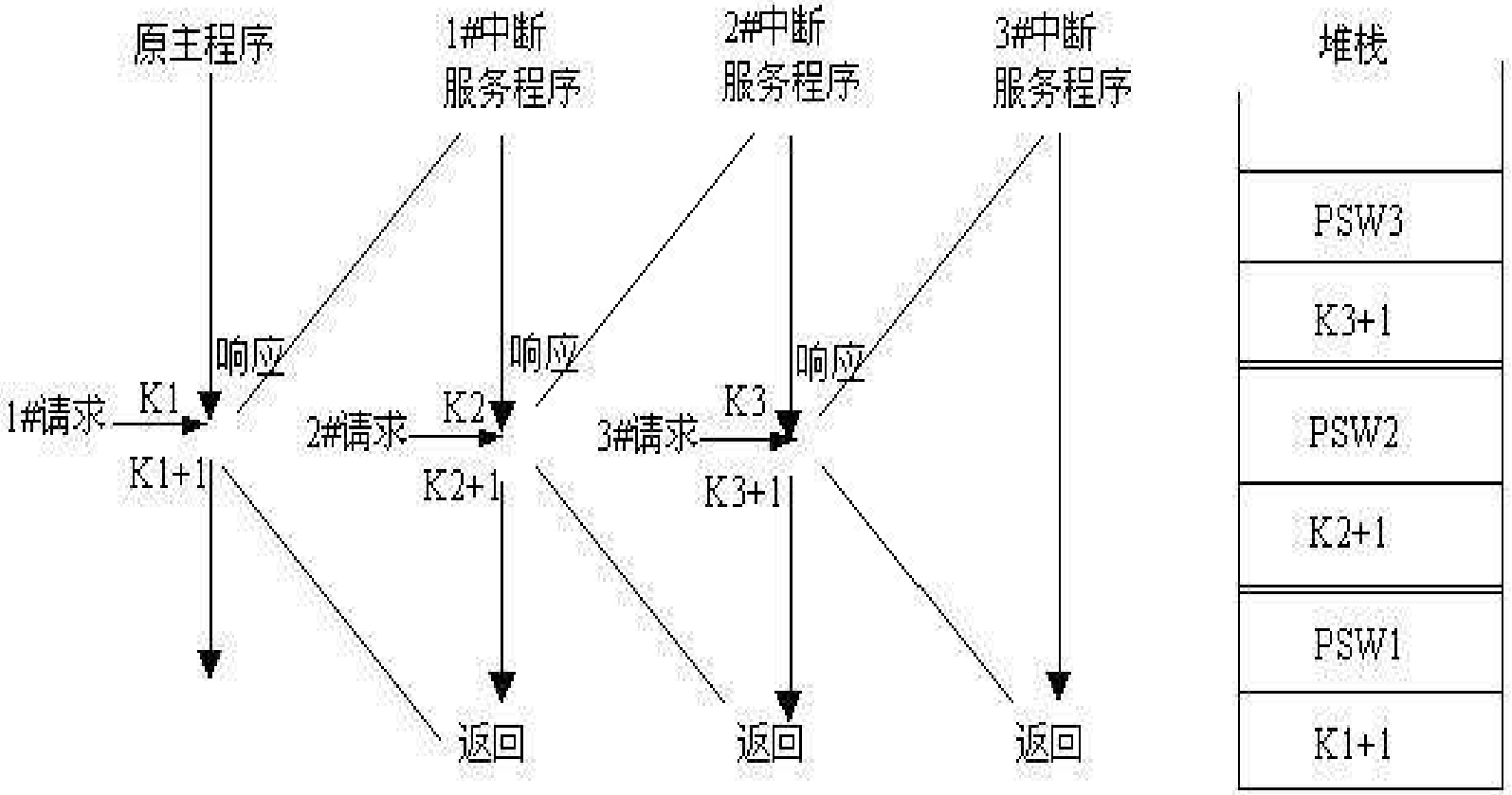
- 多重中断和中断处理优先权的动态分配
  - 多重中断的概念：

    在一个中断处理（即执行中断服务程序）过程中，若又有新的中断请求发生，而新中断优先级高于正在执行的中断，则应立即中止正在执行的中断服务程序，转取处理新的中断。这种情况为多重中断，也称中断嵌套。
  - 中断优先级的概念：

    中断响应优先级----由查询程序或硬联排队线路决定的优先权，反映多个中断同时请求时选择哪个响应。

    中断处理优先级-----由各自的中断屏蔽字来动态设定，反映本中断与其它中断间的关系。

多重中断嵌套



中断优先级的顺序是：

3# > 2# > 1#

Stack是内存中采用“FILO”的一块特殊的存储区

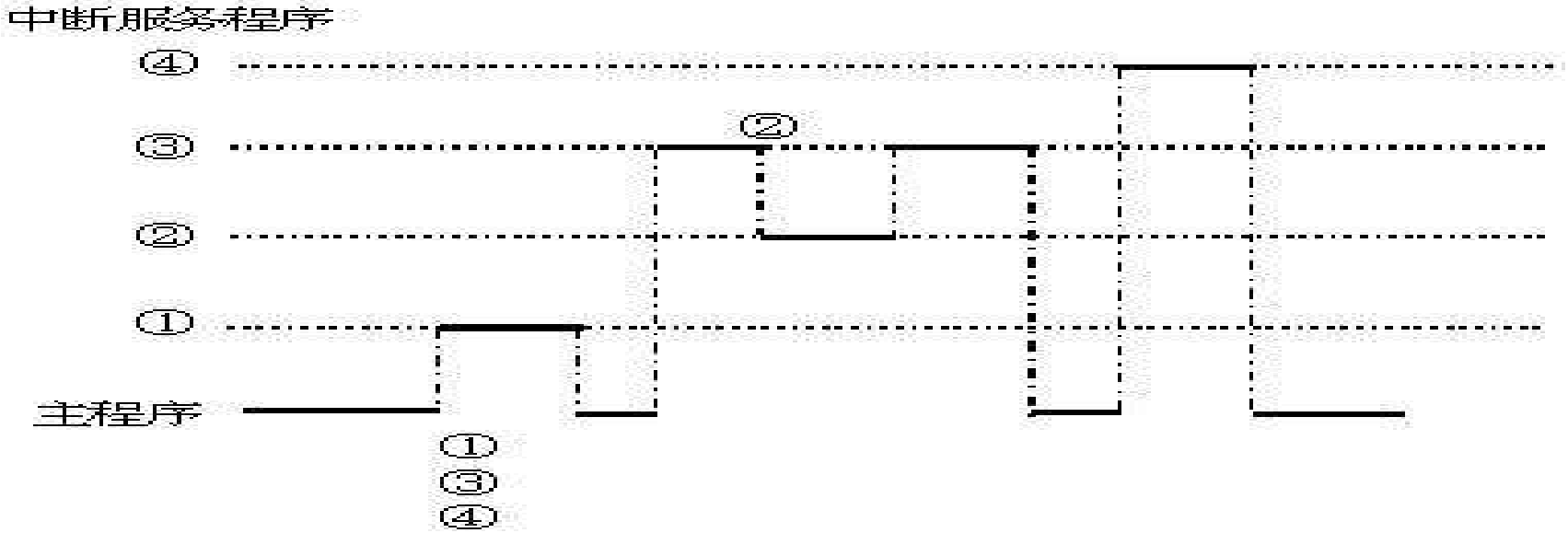
中断优先权的动态分配

- 举例：假定某中断系统有四个中断源，其响应优先级为1>2>3>4，分别写出处理优先级为1>2>3>4和1>4>3>2时各中断的屏蔽字及CPU完成中断处理的过程。

(1)中断处理优先级为1>2>3>4时：

中断程序级别	屏蔽字			
	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

（假定 1 是屏蔽，0 是开放）



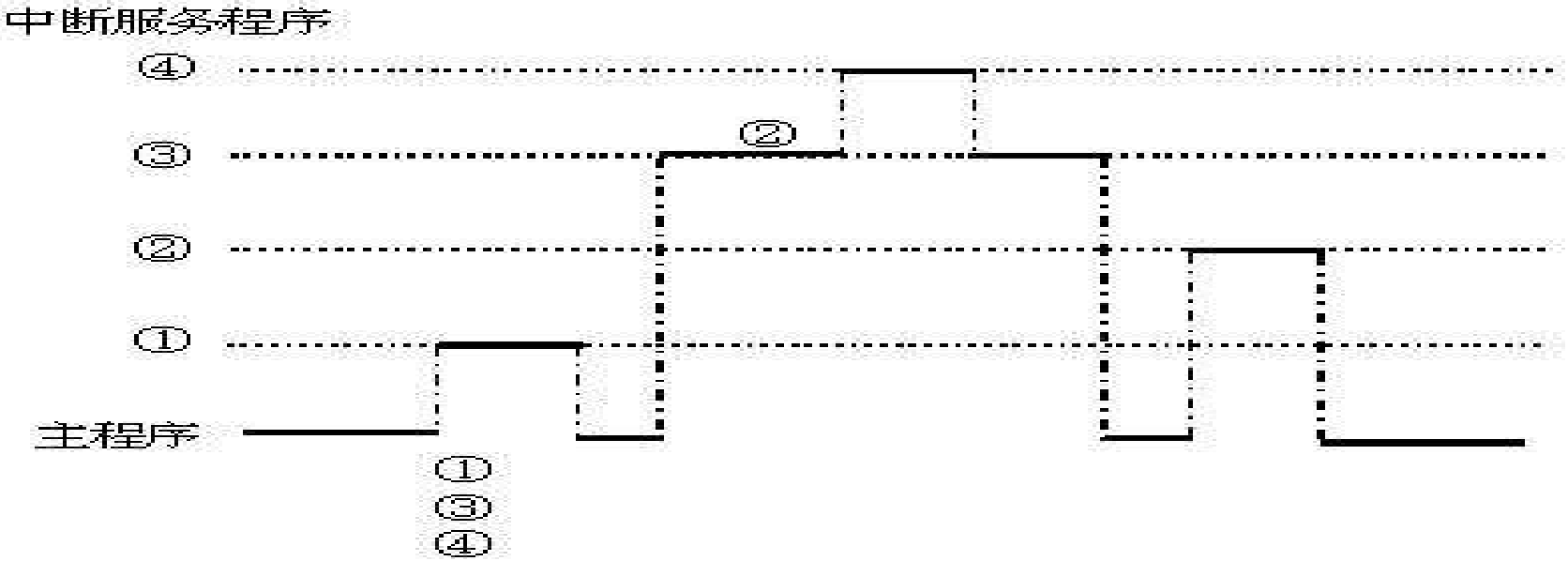
中断优先权的动态分配

- 举例：假定某中断系统有四个中断源，其响应优先级为1>2>3>4，分别写出处理优先级为1>2>3>4和1>4>3>2时各中断的屏蔽字及CPU完成中断处理的过程。

(1)中断处理优先级为1>4>3>2时：

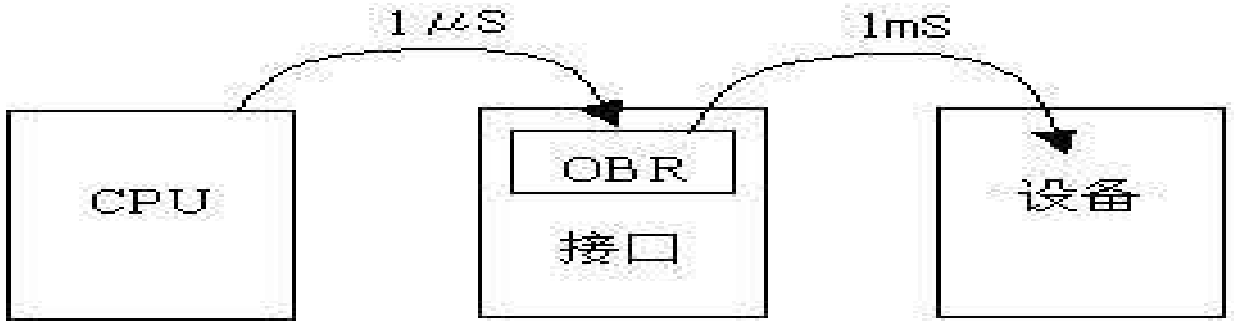
中断程序级别	屏蔽字			
	1级	2级	3级	4级
第1级	1	1	1	1
第2级	0	1	0	0
第3级	0	1	1	0
第4级	0	1	1	1

（假定1是屏蔽，0是开放）



轮询方式和中断方式的比较

- 举例：假定某机控制一台设备输出一批数据。数据由主机输出到接口的数据缓冲器**OBR**，需要**1μs**。再由**OBR**输出到设备，需要**1ms**。设一条指令的执行时间为**1μs(**包括隐指令**)**。试计算采用程序传送方式和中断传送方式的数据传输速度和对主机的占用率。



对主机占用率：

在进行**I/O**操作过程中，处理器有多少时间花费在输入/出操作上。

数据传送速度（吞吐量、**I/O**带宽）：

单位时间内传送的数据量。　　假定每个数据的传送都要重新启动！

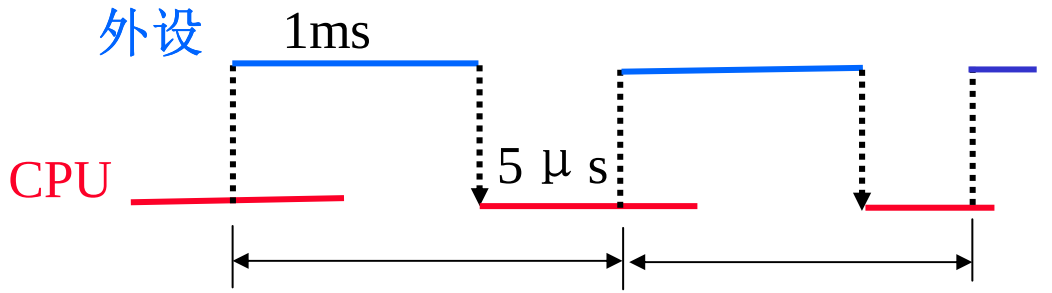
轮询方式和中断方式的比较

（1）程序直接控制传送方式

若查询程序有10条，第5条为启动设备的指令，则：

数据传输率为：1/(1000+5) μs，约为每秒995个数据。

主机占用率=100%



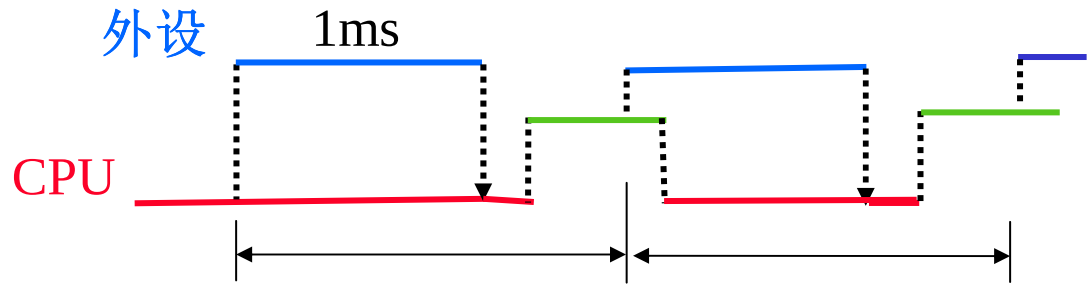
程序传送方式

（2）中断传送方式

若中断服务程序有30条，在第20条启动设备，则：

数据传输率为：1/(1000+1+20) μs，约为每秒979个数据。

主机占用率为：(1+30)/(1000+1+20)=3%



中断传送方式

问题：为什么中断服务程序比查询程序长？

因为有额外开销，如：保存现场、保存旧屏蔽字、开中断、（查询中断源）等

DMA输入/出方式

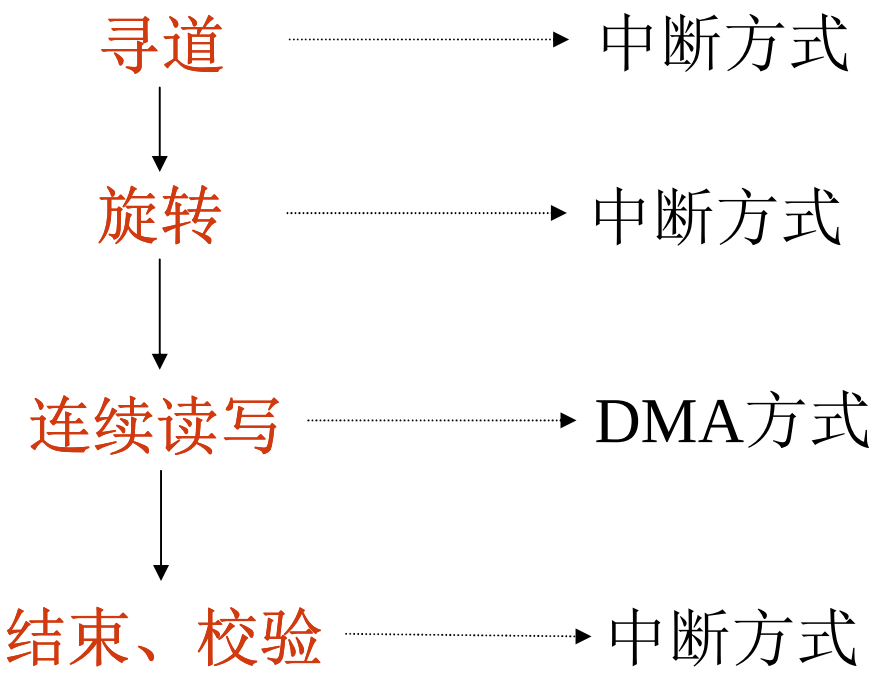
- DMA的全称
  - 直接存储器存取（Direct Memory Access）
- 为什么要引入DMA方式？
  - 程序直接控制方式受“踏步”现象的限制，效率低下，不适合高速设备和主机间的数据传送。
  - 中断控制方式虽比程序直接控制方式有效，CPU和外设有一定的并行度，但由于下列原因也不适合高速设备和主机间的数据传送。
    - 对I/O请求响应慢。每传送一个数据都要等待外设的中断请求，并增加许多中断响应和中断处理前、后的附加开销（保护断点、现场等），不能及时响应I/O请求。
    - 数据传送速度慢。数据传送由软件完成（由CPU执行相应的中断服务程序来完成)，速度慢 。

DMA方式的基本要点

- DMA方式的基本思想
  - 在高速外设和主存间直接传送数据
  - 由专门硬件（即：DMA接口）控制总线进行传输
- DMA方式适用场合
  - 高速设备（如：磁盘、光盘等）
  - 成批数据交换，且数据间间隔时间短，一旦启动，数据连续读写
- 采用“请求-响应”方式
  - 每当高速设备准备好数据，就进行一次“DMA请求”，DMA控制器接受到DMA请求后，申请总线使用权
  - DMA控制器的总线使用优先级比CPU高，为什么？
- 与中断控制方式结合使用
  - DMA传送前，“寻道”“旋转”等操作结束时，通过“中断”告知CPU
  - 在DMA控制器控制总线进行数据传送时，CPU执行其他程序
  - DMA传送结束时，要通过“DMA结束中断”告知CPU

与中断控制方式结合使用

- 举例：用于磁盘和主存间数据交换时



DMA数据传送方式

由于DMA接口和CPU共享主存，所以可能出现两者争用主存的现象，为使两者协调使用主存，DMA通常采用以下三种方式进行数据传送。

(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。

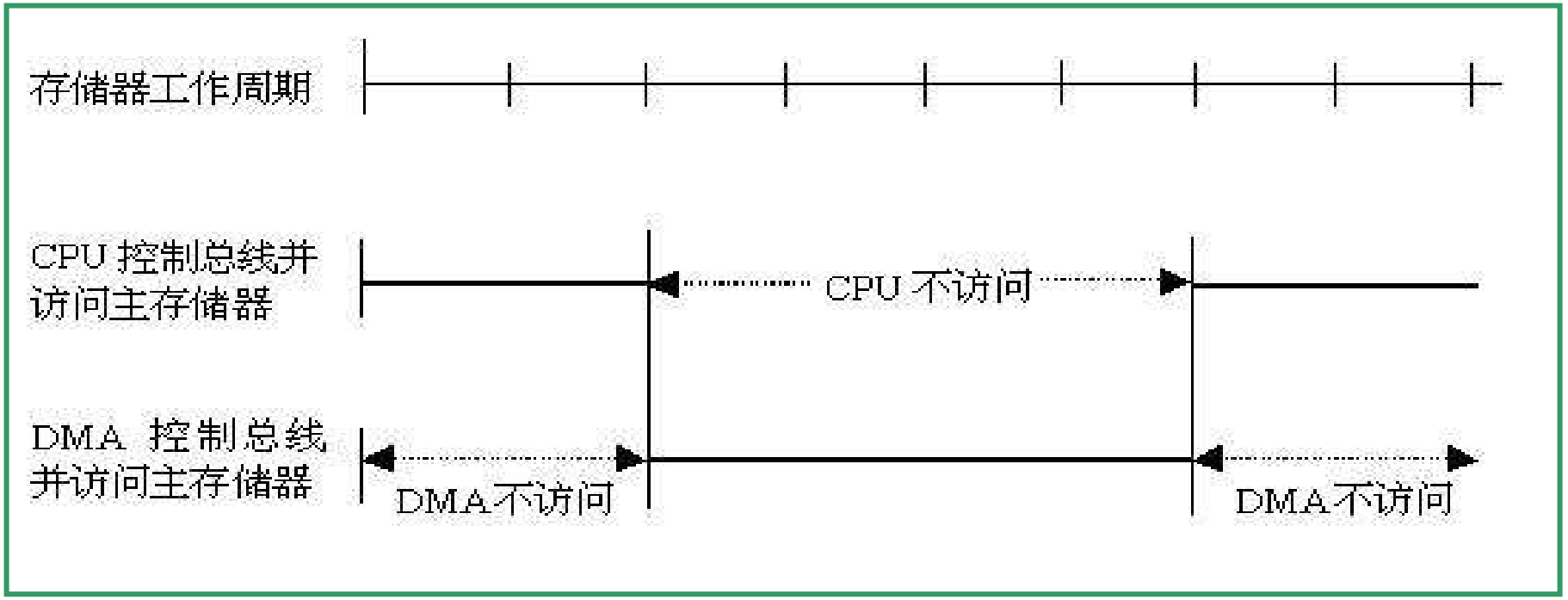
(2) 周期挪用(窃取)法(单字传送)

DMA传输时，CPU让出一个总线事务周期，由DMA控制总线来访问主存，传送完一个数据后立即释放总线。

(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，这样在每个存储周期内，CPU和DMA都可访问存储器。

CPU停止法



优点：控制简单、适用于传输率很高的外设实现成组数据传送。

缺点：CPU工作受影响。DMA访存时CPU基本上处于停止状态。主存周期没有被充分利用。即使I/O设备高速运行，但两个数据之间的准备间隔时间也总大于一个存储周期，所以主存周期没有被充分利用。

CPU停止法

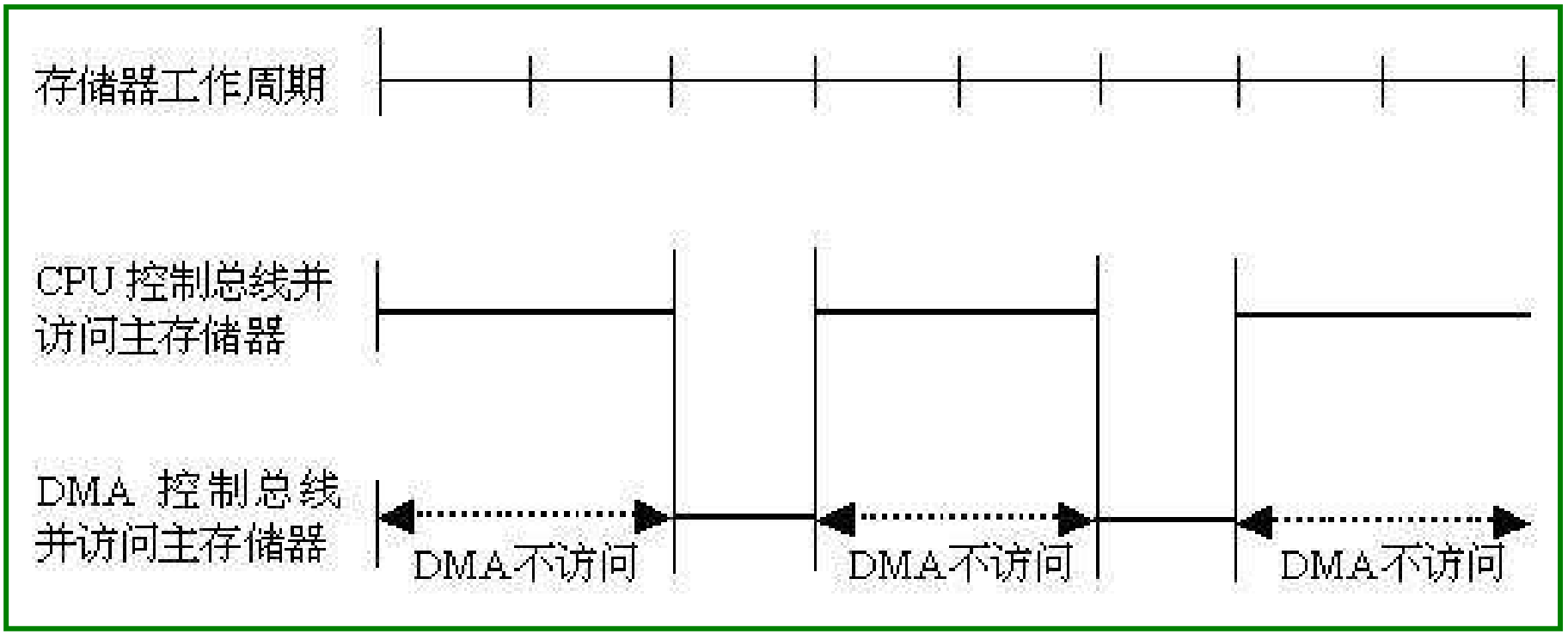
- 弥补CPU停止法缺点的做法：
  - 在DMA接口中引入缓冲器

    在DMA接口中采用一个小容量的半导体存储器，使I/O设备先和这个小容量存储器交换数据，然后再使用总线由小容量存储器与主存进行数据交换。这样可减少DMA传送数据时占用总线的时间，也就减少了CPU的等待时间。
  - 采用周期挪用（窃取）法

    挪用一个主存的存储周期进行外设和主存的一个数据交换。

    每次DMA传送完一个数据就释放总线，使在外设准备下一数据时，CPU能插空访问主存。

周期挪用法



优点：既能及时响应I/O请求，又能较好地发挥CPU和主存的效率。这种方式下，在下一数据的准备阶段，主存周期被CPU充分利用。因此适合于I/O设备的读写周期大于主存周期的情况。

缺点：每次DMA访存都要申请总线控制权、占用总线进行传送、释放总线，因此，会增加传输开销。

周期挪用法

- I/O设备要求DMA传送时可能会遇到以下三种情况之一：
  - **CPU不需访问主存**

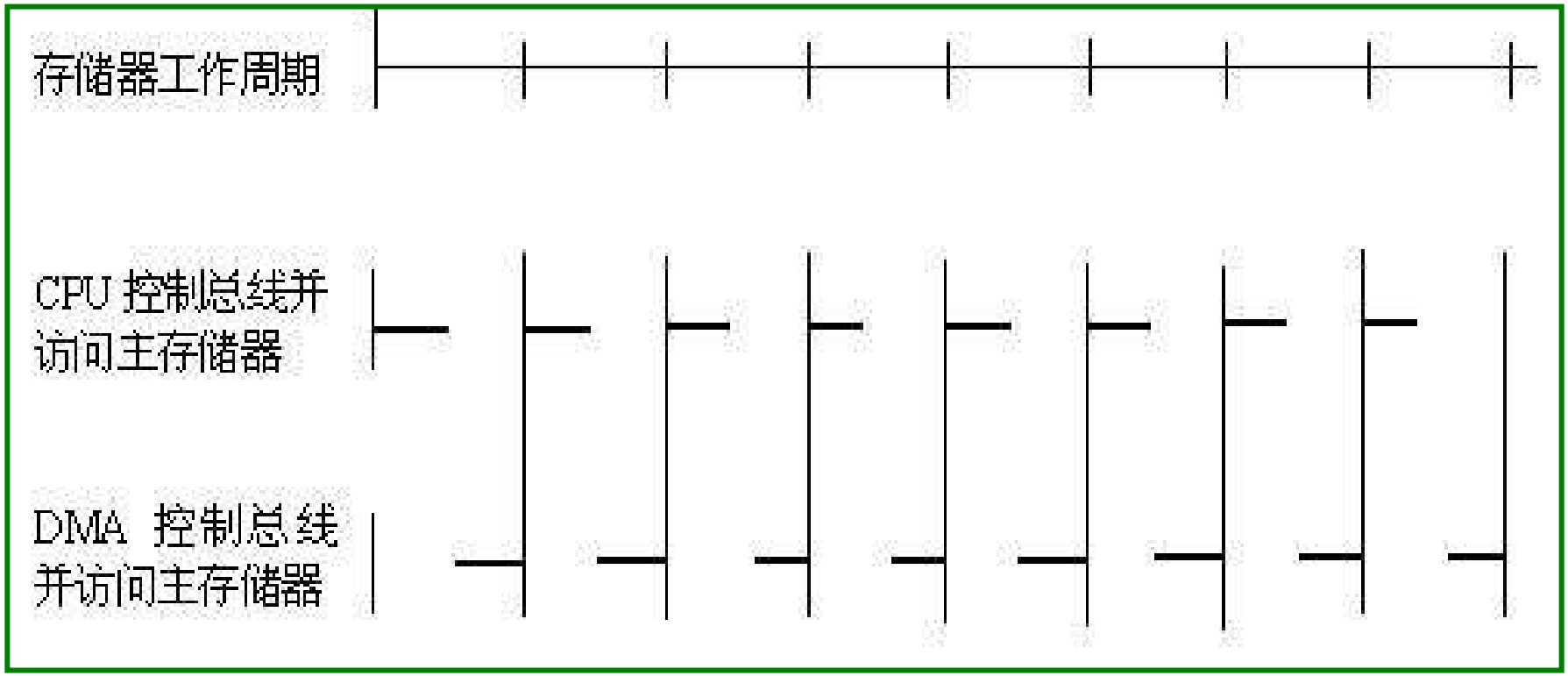
    此时，不会发生冲突，两者并行。

    如: CPU正在执行乘法指令，要花很长时间而不需马上访存。
  - **CPU正在访问主存**

    此时须等到存储周期结束，CPU让出总线，DMA才能访存。
  - **CPU也同时要访问主存**

    此时出现访存冲突。因为不马上响应DMA请求的话，高速设备可能会发生数据丢失，所以，DMA的总线优先权比CPU高。这时，先让DMA占用总线，窃取一个主存周期，完成一个数据的交换。这样，CPU便要延迟一段时间才能访存。

交替分时访问法



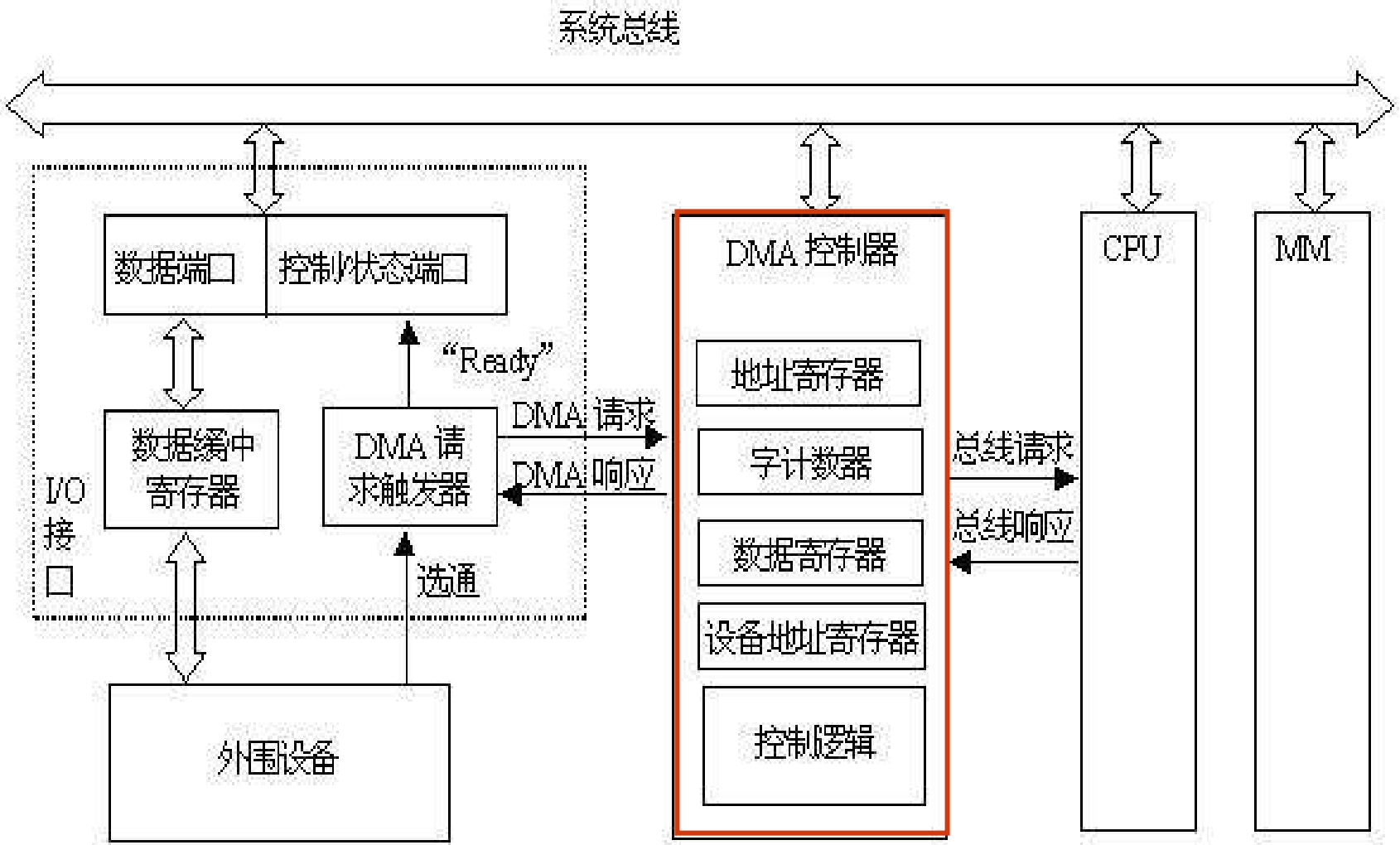
特点：适用于CPU工作周期比主存存取周期更长的情况。

不需要总线使用权的申请和释放。

在这种方式下，CPU既不停止主程序运行也不进入等待状态，在CPU工作过程中，不知不觉完成DMA数据传送，故又被称为“透明DMA”方式。

DMA方式结构

- DMA方式下的系统逻辑结构



DMA控制器的功能

DMA数据传送过程由DMA接口的控制逻辑完成，所以DMA 接口也称DMA控制器。其功能为：

(1) 请求。能接收外设发来的“DMA请求”信号，并能向CPU发“总线请求”信号。

(2) 响应。当CPU发出“总线响应”信号响应请求后，能接管对总线的控制。

(3) 修改主存地址。能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。能识别传送方向以在控制线上给出正确的读写控制信息。

(5) 确定传送数据个数。

(6) 能发出DMA结束信号。引起一次DMA中断，进行数据校验等一些后处理。

DMA控制器的操作步骤

- DMA操作步骤

  (1) DMA控制器的预置(初始化)----软件实现

  - 准备内存
  - 设置参数
  - 启动外设

  (2) DMA数据传送----硬件实现

  - **DMA请求：选通-〉DMA请求-〉总线请求**
  - **DMA响应：总线响应(CPU让出总线)-〉DMA响应**
  - **DMA传送：DMA控制总线进行数据传送**

  (3) DMA结束处理----软件实现

  根据计数值为“0”，发出DMA结束信号去接口控制产生DMA中断请求信号，转入中断服务程序，做一些数据校验等后处理工作。

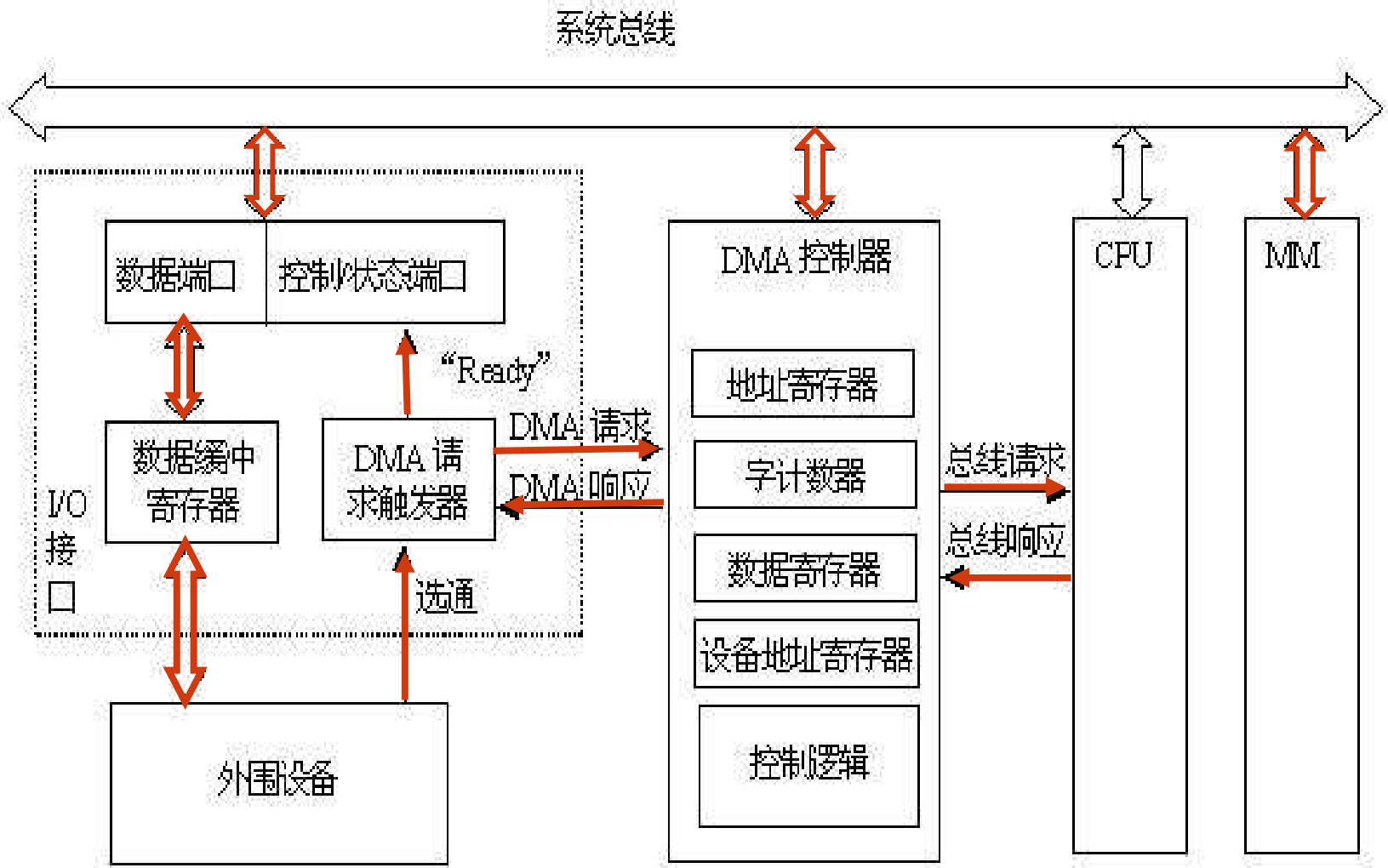
DMA控制器的初始化

- DMA控制器的预置(初始化)
  - 准备内存区
    - * 输入：在内存设置好缓冲区
    - * 输出：先在内存准备好数据
  - 设置传送参数

    执行I/O指令，测试外设状态，对DMA控制器设置各种参数：
    - * 内存首址=〉地址寄存器
    - * 字计数值=〉字计数器
    - * 传送方向=〉控制寄存器
    - * 设备地址=〉设备地址寄存器
  - 启动外设，然后执行其他程序

DMA传输过程

- DMA的结构



DMA传输过程

(1) 当外设准备好数据，或准备好接收数据时，发“选通”信号，使数据送数据缓冲寄存器，同时DMA请求触发器置“1”。

(2) DMA请求触发器向控制/状态端口发“Ready”信号，同时向DMA控制器发“DMA请求”信号。

(3) DMA控制器向CPU发“总线请求”信号。

(4) CPU完成现行机器周期后，响应DMA请求，发出“总线响应”信号。DMA控制器接受到该信号后，发出“DMA响应”信号，使DMA请求触发器复位。此时，CPU浮动它的总线，让出总线控制权，由DMA控制器控制总线。

(5) DMA控制器给出内存地址，并在其读/写线上发出“读”或“写”命令，随后在数据总线上给出数据。

(6) 根据读写命令，将数据总线上的数据写入存储器中，或写入数据端口，并进行主存地址增量，计数值减1，

若采用“CPU停止法”，则循环第6步，直到计数值为“0”。

若采用“周期挪用法”，则释放总线（下次数据传送时再按过程(1)到(6)进行）。

DMA方式和中断方式的区别

(1) DMA方式下数据传送由硬件(DMA控制器)完成；中断方式下，数据传送由软件（CPU执行中断服务程序）完成。

(2) DMA请求对存储器访问的请求，也即对总线控制权的请求，没有中止现行程序的必要；而中断请求要处理器转去执行中断服务程序，因此要中止现行程序，保存断点、现场等。

(3) 中断除了能完成外设和主机的数据交换，还能处理异常事件；而DMA方式下不能处理异常事件。

(4) 中断响应在一个指令周期结束后；而DMA响应是在一个总线周期后。

(5) DMA方式用于高速设备；而中断方式用于低、慢速设备。

(6) DMA方式下，外设与CPU并行度高；而中断方式下，外设与CPU并行度低。

例：中断、DMA方式下CPU的开销

设处理器按500MHz的速度执行，硬盘以 4 字块(每字32位)进行传送，速率为4MB/Sec，且没有任何数据传输被错过。

（1）若用中断驱动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。如果硬盘仅用5%的时间进行传送，那么处理器用在硬盘I/O操作上所花的时间百分比（主机占用率）为多少？

（2）若用DMA方式，处理器花1000个时钟进行DMA传送的初始化设置，并且在DMA完成后的中断处理需要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）,那么当硬盘进行传送的时间占100%（即：硬盘一直进行读写，并传输数据）时，处理器用在硬盘I/O操作上的时间百分比（主机占用率）为多少？

例：中断、DMA方式下CPU的开销

- 中断传送：
  - 硬盘要求每次中断以 4 字块（=16字节）进行传送，为了保证没有任何数据传输被错过，传送的速率应达到每秒4MB/16B=250k次中断的速度；
  - 每秒钟用于中断的周期数为$250kx500=125x10^6$；
  - 在一次数据传输中，处理器花费在I/O上的时间的百分比为：$125x10^6/(500x10^6)=25\%$；
  - 假定硬盘仅用其中5%的时间来传送数据，则处理器花费在I/O方面的百分比为$25\%x5\%=1.25\%$ 。
- DMA传送：
  - 每次DMA传送将花费$8KB/(4MB/Sec)\approx 2x10^{-3}$秒；
  - 一秒钟内有$1/(2x10^{-3})=500$次DMA传送；
  - 如果硬盘一直在传送数据的话，处理器必须每秒钟花$(1000+500)x500=750x10^3$个时钟周期来为硬盘I/O操作服务；
  - 在硬盘I/O操作上处理器花费的时间占：

$$750x10^3/(500x10^6)=1.5x10^{-3}=0.15\%$$ 。

通道和IOP方式

I/O方式的发展过程：

第一阶段：CPU直接控制外设。

第二阶段：增加一个控制器或一个I/O模块，CPU使用编程I/O控制。

第三阶段：采用中断技术，CPU不需要花费时间等待外设执行I/O操作，实现了外设和CPU的并行。

第四阶段：I/O模块通过DMA直接传送一块数据到或从存储器传出，不需要CPU全部参与。

大型计算机系统采用以下I/O方式：

(1) I/O模块具独立I/O处理能力，具执行专门I/O程序的能力。CPU只需在主存中事先组织好I/O程序，发出相应的I/O命令即可，对I/O的干预极少。这种方式为通道方式。

(2) I/O模块是一个专门的I/O处理器，拥有自己单独的存储器和指令集，CPU参与更少。这种方式为I/O处理器方式。

小结

- 有三种基本的I/O传输方式
  - 轮询方式（程序直接控制 / 程序查询方式）：通过查询程序定时到I/O端口取状态来查询外设和I/O控制器的状况，以控制设备进行相应的动作
  - 程序中断方式（中断驱动方式）：当外设完成任务或发生特殊情况时，由外设主动向CPU提出中断请求，CPU在每条指令结束后查询有无中断请求，有则转内核态，调出操作系统中的中断处理(服务)程序来处理外设请求。在中断处理程序中完成CPU和外设的数据传送
    - 中断响应过程（硬件-处理器）：关中断、保护断点、转中断服务程序
    - 中断服务程序的入口地址可以是一个中断查询程序入口，也可以由中断控制器给出中断类型号，再根据类型号到中断向量表中取入口地址
    - 中断处理过程（软件-OS）：准备阶段、处理阶段、恢复并返回阶段
    - 中断控制器：记录所有中断请求，在中断掩码（屏蔽码）的作用下，对所有未被屏蔽的中断请求进行优先权编码，送出优先级最高的中断类型号给CPU
    - 多重中断：在处理中断时又发生新中断请求的处理机制
  - 直接存储器访问方式（DMA方式）：用于磁盘等高速外设与主存之间直接数据交换
    - DMA控制器的结构：字节计数器、地址寄存器、设备地址寄存器、控制逻辑等
    - 三种控制方式：CPU停止法、周期挪用法、交替分时访问法
    - DMA传输过程：控制器初始化、数据传输、结束处理