计算机组成原理

> ---NonoTion

第 1章 计算机系统概论

1.1 计算机系统简介

计算机系统由软件和硬件组成

计算机系统的层次结构

高级语言机器->汇编语言机器->操作系统机器->机器语言机器->微指令系统

虚拟机器M4　虚拟机器M3 虚拟机器M2 传统机器M1 微程序机器M0

1.2 计算机的基本组成

冯·诺伊曼计算机的特点

- 计算机由运算器、存储器、控制器、输入系统和输出系统五大部件组成
- 指令和数据以同等地位存放于存储器内，并可按地址寻访
- 指令由操作码和地址码组成，操作码用来表示操作的性质，地址码用来表示操作数在存储器中的位置
- 指令在存储器内按顺序存放。通常指令是顺序执行的，在特定条件下，可根据运算结果或根据设定的条件改变执行顺序
- 机器以运算器为中心，输入输出设备与存储器之间的数据传送通过运算器完成

现代计算机以存储器为中心

各部件的功能如下

- 运算器用来完成算术运算和逻辑运算，并将运算的中间结果暂存在运算器内
- 存储器用来存放数据和程序
- 控制器用来控制、指挥程序和数据的输入、运行以及处理运算结果
- 输入设备用来将人们熟悉的信息形式转换为机器能够识别的信息形式
- 输出设备可将机器运算结果转换为人们熟悉的信息形式

现代计算机三大部分

- CPU=运算器+控制器
- 输入和输出设备简称为I/O设备
- 主存储器

1.3 计算机硬件的主要技术指标

- 机器字长——CPU一次能够处理数据的位数
- 存储容量——分为主存容量和辅存容量
- 运算速度
  - 主频
  - 吉普森法 $T_M = \sum_{i=1}^{n} f_i t_i$

  f_i第i种指令占全部操作的百分比，t_i第i种指令执行的时间
  - MIPS 每秒执行百万条指令
  - CPI 执行一条指令所需时钟周期数
  - FLOPS 每秒浮点运算次数

第2章 计算机的发展及应用

了解即可

第3章 系统总线

3.1 总线的基本概念

- 总线连接——将各部件连接到一组公共信息传输线上
- 总线——连接多个部件的信息传输线，是各部件共享的传输介质

同一时间，只有一个部件可以向总线发送信息，多个部件可以从总线上接受相同的信息

3.2 总线的分类（按不同连接部件分）

3.2.1 片内总线

芯片内部的总线 连接CPU芯片内部的部件

3.2.2 系统总线

CPU、主存、I/O设备(通过I/O接口)各大部件之间的信息传输线

按系统总线传输信息的不同，可分为三类

1. 数据总线——同于传输各功能部件之间的数据信息，双向传输总线，其位数与机器字长、存储字长有关。数据总线的位数称为**数据总线宽度**
2. 地址总线——用来指出数据总线上的源数据或目的数据在主存单元的地址或I/O设备地址,CPU输出，单向传输。
3. 控制总线——用来**发出各种控制信号**的传输线，对任一控制线而言，它的传输是单向的。对于控制总线总体来说，又可以认为是双向的。还起到**监视各部件状态**的作用

3.2.3 通信总线

用于计算机系统之间或计算机系统与其它系统之间的通信

串行通信和并行通信

3.3 总线特性及性能指标

3.3.1 总线特征

- 机械特性——总线在机械连接方式上的一些性能
- 电气特性——总线的每一根传输线上信号的传递方向和有效的电平范围
- 功能特性——总线中每根传输线的功能
- 时间特性——总线的任一根线在什么时间内有效。一般用信号时序图来描述

3.3.2 总线性能指标

- 总线宽度 ——通常是指数据总线的根数，用bit(位)表示
- 总线带宽——总线的数据传输速率
- 时钟同步/异步——总线上的数据与时钟同步工作的总线叫做同步总线，与时钟不同步工作的总线叫做异步总线
- 总线复用——地址总线、数据总线和控制总线三种总线数的总和
- 总线控制方式——包括突发工作、自动配置、仲裁方式、逻辑方式、计数方式等
- 其他——如负载电压、电源电压等

3.3.3 总线标准

系统与各模块、模块与模块之间的一个互连的标准界面

如：ISA、EISA、VESA、PCI...

3.4 总线结构

3.4.1 单总线结构

将CPU、主存、I/O设备(通过I/O接口)挂在一组总线上，允许I/O设备之间,I/O设备与CPU之间或I/O设备与主存之间直接交换信息

优点：简单，便于扩充

缺点：影响系统工作效率的提高

3.4.2 多总线结构

双总线结构

将速度较低的I/O设备从但总线上分离出来、形成主存总线与I/O总线分开的结构

三总线结构

将速率不同的I/O设备进行分类，然后将它们连接在不同的通道上，那么计算机系统的工作效率将会更高

四总线结构

3.5 总线控制

3.5.1 总线判优控制

总线上连接的各种设备，按其对总线有无控制功能可分为主设备(模块)和从设备(模块)、

多个主设备同时要使用总线时，要按一定的优先级顺序决定哪个设备能使用总线

总线判优控制可分为两种:

集中式

- 链式查询——控制总线中有三根线用于总线控制(BS总线忙,BR总线请求,BG总线同意)。优点：几根线就能按一定优先次序实现总线控制 缺点：对电路故障很敏感
- 计数器定时查询——多了一组设备地址线，少了一根总线同意线
- 独立请求方式——每一台设备均有一对总线请求线BR和总线同意线BG

3.5.2 总线通信控制

通常完成一次总线操作的时间称作总线周期，可分为下面四个阶段

- 申请分配阶段——需要使用总线的主模块提出申请，经总线仲裁机构决定下一传输周期的总线使用授权于某一申请者
- 寻址阶段——主模块通过总线发出本次要访问的从模块的地址及有关命令
- 传数阶段——主模块和从模块进行数据交换
- 结束阶段——主模块的有关信息从系统总线上撤出，让出总线使用权

四种方式

1.同步通信 **重点**

通信双方由统一时标控制数据传送称为同步通信

2.异步传输

(1)不互锁方式

主模块发出请求信号后，不必等到接到从模块的回答信号，而是经过一段时间，确定从模块已收到请求信号后，撤销其请求信号；从模块接到请求信号后，在条件允许时发出回答信号且经过一段时间确定主模块已收到回答信号后，自动撤销回答信号

(2)半互锁方式

主模块发出请求信号后，必须等到接到从模块的回答信号后，再撤销其请求信号

(3)全互锁方式

主模块发出请求信号后，必须等到接到从模块的回答信号后，再撤销其请求信号。从模块接到请求信号后，必须待获知主模块请求信号撤销后，再撤销其回答信号

异步串行通信的数据传输率用波特率来衡量。

波特率——单位时间内传送二进制位数

3.半同步通信

既保留了同步通信的基本特点，同时像异步通信意义，允许不同速度的模块进行工作，为此增加了一条等待响应线（\WAIT）

4.分离式通信

将一个传输周期(或总线周期)分解为两个子周期

第4章 存储器

4.1 概述

4.1.1 存储器分类

按存储介质分类

- 半导体存储器——易失性存储器
- 磁表面存储器
- 磁芯存储器
- 光盘存储器

按存取方式分类

- 随机存储器 RAM
- 只读存储器 ROM
- 串行访问存储器 SAM

按再计算机中的作用分类

- 主存储器
- 辅助存储器
- 缓冲存储器
- 闪速存储器

4.1.2 存储器的层次结构

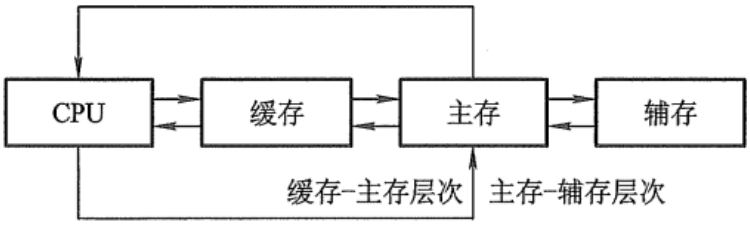


图 4.3　缓存-主存层次和主存-辅存层次

CPU和缓存、主存都能直接交换信息

缓存-主存层次 解决CPU和主存速度不匹配的问题

主存-辅存层次 解决存储系统的容量问题

4.2 主存储器

4.2.1 概述

主存中存储单元地址的分配

主存各存储单元的空间位置是由单元地址号来表示的，而地址总线是用来指出存储单元地址号的，根据该地址可以读出或写入一个存储字

通常计算机系统可以按字寻址，也可以按字节寻址

主存的技术指标

- 存储容量——主存能够存放二进制代码的总位数

$$存储容量 = 存储单元个数 * 存储字长$$

  存储字长——存储单元中的二进制位数 一般取8的倍数
- 存储速度——由存取时间和存取周期来表示
  - 存取时间——启动一次存储器操作到完成该操作所需的全部时间
  - 存取周期——存储器连续进行两次独立的存储器操作所需的最小时间间隔
- 存储器带宽——单位时间内存储器存取的数据量

提高存储器带宽可采取一下措施

- 缩短存取周期
- 增加存储字长
- 怎加存储体

4.2.2 半导体存储芯片简介

1.半导体存储芯片的基本结构

- 译码驱动——把地址线送来的地址信号翻译位对应的存储单元的选择信号

- 读/写电路——包括放大器和写入电路，用来完成读写操作
- 存储芯片——通过地址总线、数据总线、控制总线与外部连接

地址线是单向输入的，其位数与芯片容量有关

数据线是双向的，其位数与芯片可以写入或读出的数据位数有关

地址线和数据线共同反映存储芯片的容量

控制线

- 读/写控制线 决定芯片进行读/写操作
- 片选线 用来选择存储芯片

2.半导体存储芯片的译码驱动方式

线选法

只用一根字选择线，直接选中一个存储单元的各位

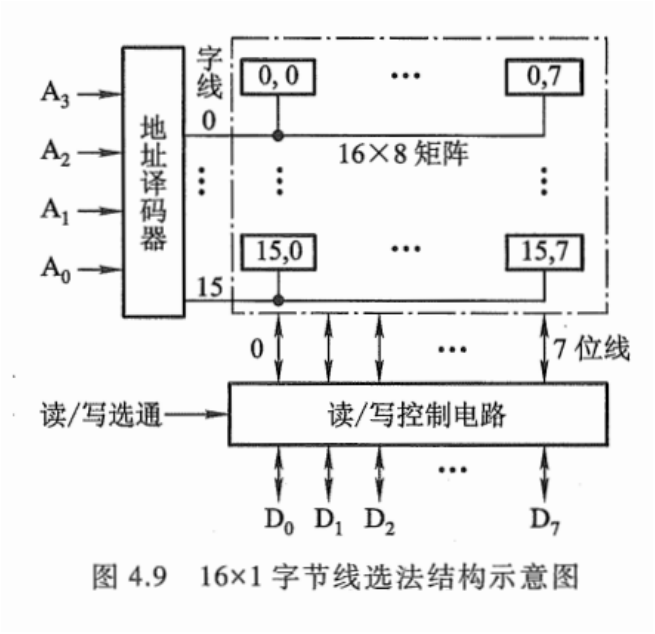


图 4.9　16×1 字节线选法结构示意图

优点：结构简单

缺点：只适用于容量不大的存储芯片

重合法

被选单元是由X、Y两个方向的地址线决定的

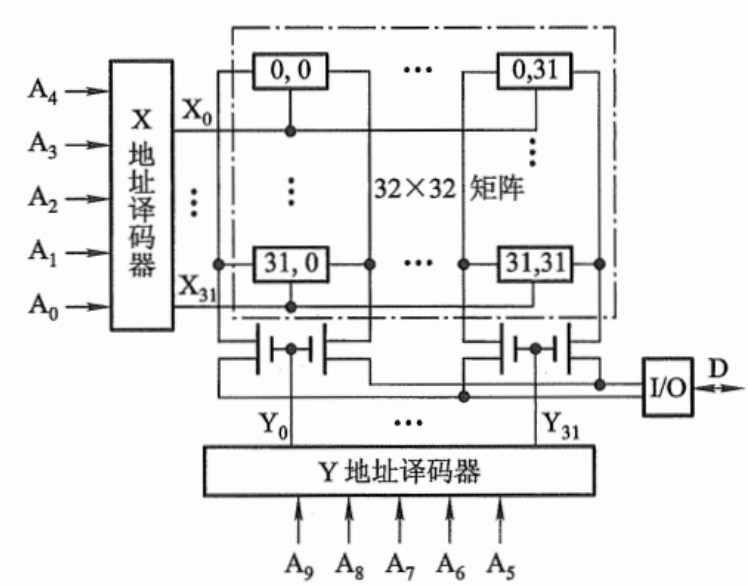


图 4.10　1 K×1 位重合法结构示意图

4.2.2 随机存取存储器

随机存取存储器按其存储信息的原理不同，可分为静态RAM和动态RAM两大类

静态RAM SRAM

触发器工作原理存储信息，因此信息被读出后，它仍保持其原状态，不需要再生。但电源掉电时，原存信息丢失，故它属易失性半导体存储器

Cache

动态RAM DRAM

有三管式和单管式两种，靠电容存储电荷的原理来寄存信息。必须进行再生或刷新

读出是破坏性读出，必须再生

比静态RAM相比，集成度更高，能耗更低

DRAM的刷新

将原存信息读出，再由刷新放大器形成原信息并重新写入的再生过程

- 集中刷新——在一个刷新周期内，对全部存储单元集中一段时间逐行进行刷新。会产生死时间或死区
- 分散刷新——对每行存储单元的刷新分散到每个存取周期内完成 存取周期变长，整个系统速度变慢

异步刷新——前两种方式的结合，缩短死时间，又充分利用最大刷新间隔2ms

动态RAM与静态RAM的比较

- 在相同大小的芯片种，动态RAM集成度远高于静态RAM
- 动态RAM的行、列地址按先后顺序输送，减少了芯片引脚和封装尺寸
- 动态RAM功耗比静态RAM小

- 动态RAM价格比静态RAM低
- 速度比SRAM低
- 动态RAM需要再生，故需配置再生电路，也需要消耗一部分功率

4.2.4 只读存储器

- 掩膜ROM 用户无法改变原始状态
- PROM 一次性编程的只读存储器
- EPROM 可擦除可编程只读存储器

4.2.5 存储器和CPU的连接

1. 存储容量的扩展

- 位扩展指的是怎加存储字长
- 字扩展指的是怎加存储器字的数量
- 字、位扩展

2. 存储器与CPU的连接

（1）地址线的连接

CPU地址线往往比存储芯片的地址线多，通常将CPU地址线的低位与存储芯片的地址线相连。CPU地址线的高位或在存储芯片时用，或做其他用途，如片选信号等

（2）数据线的连接

CPU的数据线与存储芯片数据线也不一定相等，所以必须对存储芯片进行位扩展

（3）读/写控制线的连接

（4）片选线的连接

（5）合理选择存储芯片

通常选用ROM存放系统程序，标准子程序和各类常数等。RAM则是为用户编程而设置的

4.2.6 存储器的校验

1.汉明码的组成

设欲检测的二进制代码位n位，为了使其具有纠错能力，需添加k位检测位，组成n+k位的代码。为了能准确对错误定位以及指出代码没错，新增添的检测位数k满足

$$2^k \geq n + k + 1$$

将k位检测位分别安排在代码的2^{k-1}位上，记作$C_i(i = 1, 2, 4, 8...)$

C_1检测的g_1小组包含1，3，5，7，$9...$位

C_2检测的g_2小组包含$2, 3, 6, 7, 10...$位

以此类推

…

具体规则参照P101页

2.汉明码的纠错过程

对传送后的汉明码形成新的检测位$P_i(i=1,2,4,8...)$

根据P_i的状态就可以直接指出错误的位置

$P_i=g_i$小组的所有数异或

将检测位P_i从大到小排列组成的二进制数的大小就是出错的位数（偶校验）

4.2.7 提高访存速度的措施

除了寻找高速原件和采用层次结构外，**调整主存的结构**也可提高访存速度

1.单体多字系统

由于程序和数据在存储体内是连续存放的，因此CPU访存取出的信息也是连续的，如果可以在一个存取周期内，从同一地址连续取出4条指令，然后再逐条将指令送至CPU执行，即每隔1/4存取周期，主存向CPU送一条指令，这样显然增大了存储器的带宽，提高了单体存储器的工作速度

使用这种方法的前提是：指令和数据再主存内必须是连续存放的，一旦遇到转移指令，或操作数不能连续存放，这种方法的效果就不明显

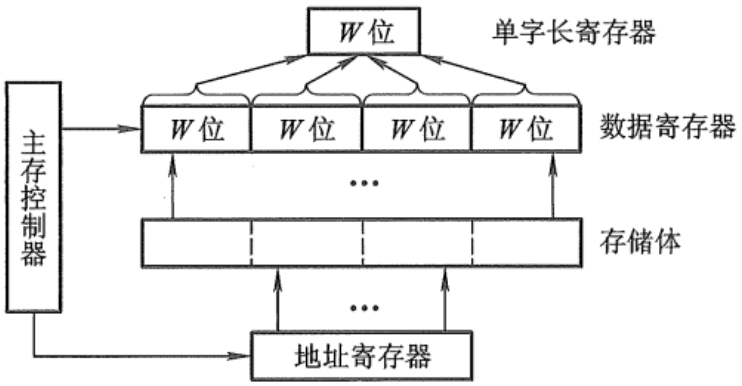


图 4.41　单体四字结构存储器

2.多体并行系统

多体并行系统就是采用多体模块组成的存储器。每个模块具有相同的容量和存取速度，各模块各自都有独立的地址寄存器(MAR)、数据寄存器(MDR)、地址译码、驱动电路和读/写电路，它们能并行工作，不能交叉工作

高位交叉编址的多体存储器

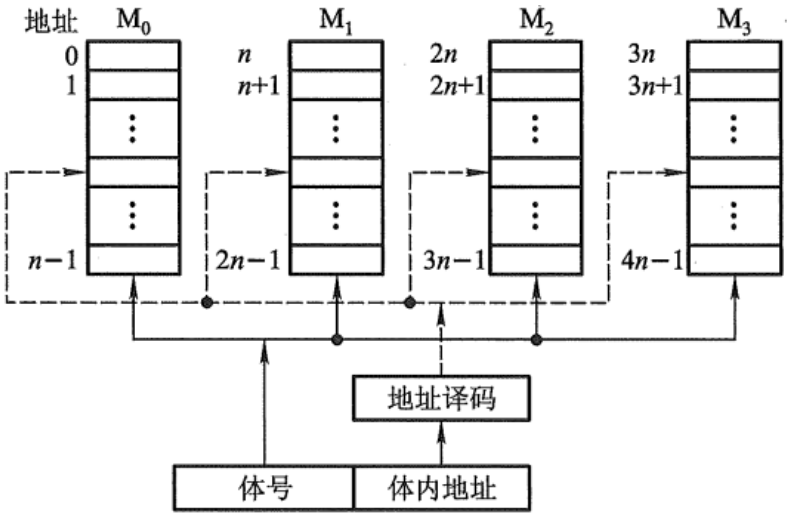


图 4.42　高位交叉编址的多体存储器

高位地址可表示体号，低位地址为体内地址

程序按照体内地址顺序存放

只要合理调动，使不同的请求源同时访问不同的体，便可实现并行工作

低位交叉编址的多体存储器

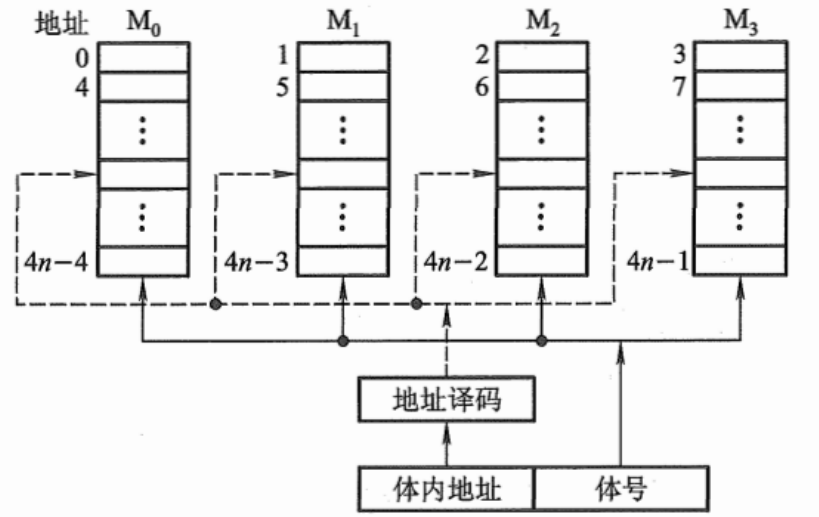


图 4.43　低位交叉编址的多体存储器

由于程序连续存放在相邻体种，故又有交叉存储之称。显然低位地址用来表示体号，高位地址为体内地址

这种编址方式又叫做模M编址（M为模块数）

多体模块结构的存储器采用交叉编址后，可以再不改变模块存取周期的情况下，提高存储器的带宽

假设每个个体的存储字长和数据总线的宽度一致，地位交叉的存储器模块数为n，存取周期为T，总线传输周期为r，采取流水线方式存取时，应满足T=nr

- 采用低位交叉编址时，t=T+(n-1)r
- 采用高位交叉编址时，t=nT

多体模块存储器不仅要与CPU交换信息，也要与辅存，I/O设备等交换信息所以需要安排各部件访问的顺序

这个部件叫做存储器控制部件

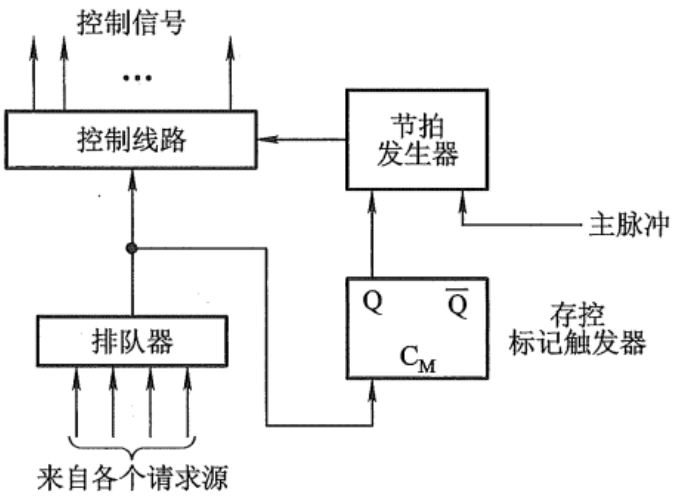


图 4.46　存控基本结构框图

有排队器、存控标记触发器、节拍发生器和控制线路组成

3.高性能存储芯片

SDRAM 同步DRAM

RDRAM

CDRAM 带Cache的DRAM

4.3 高速缓冲存储器

4.3.1 概述

1. 问题的提出

大概了解为什么要设置高速缓冲存储器P109

大体概括为两点

- 多体并行存储系统中，I/O设备访问主存的权限大于CPU，又是CPU甚至要等待几个周期才能访问主存
- 主存的速度提高始终跟不上CPU的发展

Cache的出现使CPU可以不直接访问主存，而与高速Cache交换信息

2. Cache工作原理

主存由2^n个可编制的字组成，每一字有唯一的n位地址。为了与Cache映射，将主存与缓存部分都分成若干的块，每个块内包含若干个字，并使它们的块大小相同。这就使得主存的地址分成主存块号和块内地址两部分，

缓存也分为两端：块号和块内地址

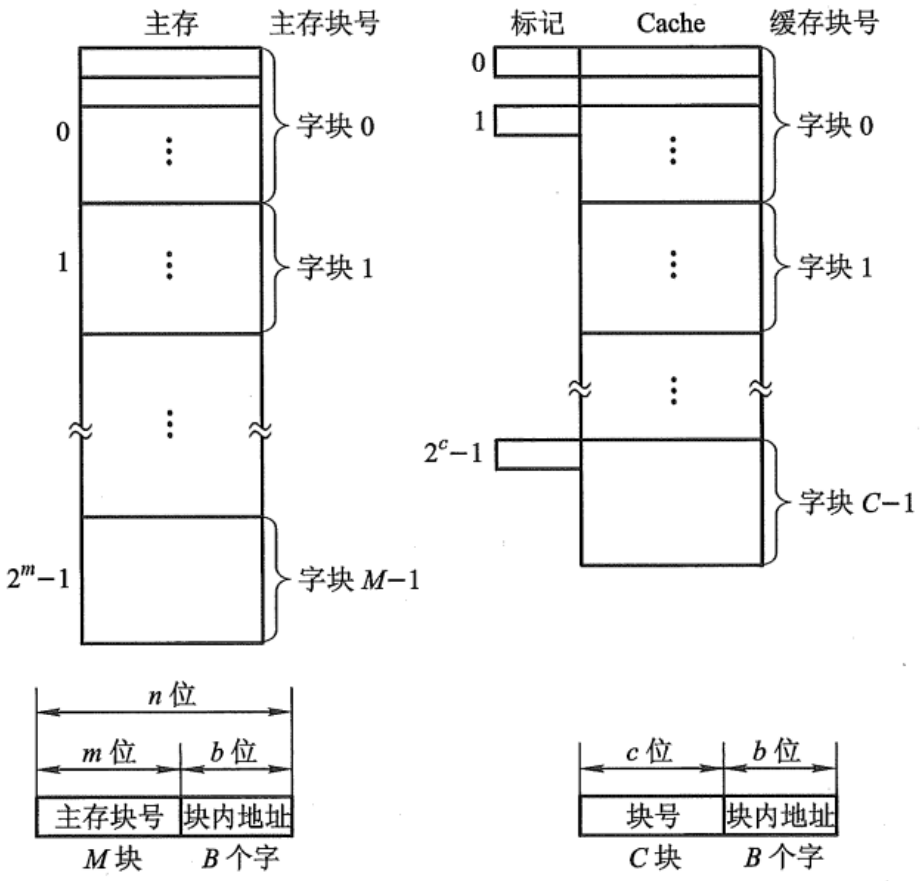


图 4.49　Cache-主存存储空间的基本结构

Cache命中与Cache不命中

- Cache命中——CPU欲读取主存某字时，所需要的字已在缓存中，即可直接访问Cache
- Cache不命中——所需的字不在缓存中，这时需要将该字所在的主存整个子块一次调入Cache中

Cache命中率

N_c为访问$Cache$的命中总次数，N_m是访问主存的总次数，则命中率h

$$h = \frac{N_c}{N_c + N_m}$$

Cache—主存系统平均访问速度

t_c为$Cache$访问时间，t_m为未命中时的主存访问时间，$1-h$表示为命中率，则平均访问时间t_a

$$t_a = ht_c + (1-h)t_m$$

访问效率e

$$e = \frac{t_c}{t_a}$$

3. Cache基本结构

- Cache存储体——Cache存储体以块为单位与主存交换信息，为加速Cache与主存之间的访问速度，主存大多采用多体结构，且Cache访存优先级最高

- 地址映射变换机构——将CPU送来的主存地址转换为Cache地址，主要是主存块号与Cache块号间的转换
- 替换机构——当Cache内容已满，无法接受来自主存块的信息时，就由Cache内的替换机构按一定的替换算法来确定应从Cache内溢出哪个块来返回主存，而把新的主存块调入Cache
- Cache读写操作

读操作：CPU发出主存地址后，首先判断存储字是否在Cache内，若命中，直接访问Cache，若不命中，一方面要访问主存，将该字传送给CPU，也要将该字所在的主存块装入Cache，若Cache已满，则要执行替换算法

写操作：

- 写直达法：写操作时数据既写入Cache又写入主存
- 写回法：写操作时数据只写入Cache，当Cache数据被替换出去时才写回主存

为了识别Cache中的数据是否与主存中一致，要增加一个标志位"清" "浊"

”清“表示未修改Cache，这时不必写回主存

”浊“表示修改Cache，这时要写回主存

4.Cache的改进

（1）单一缓存和多级缓存

（2）统一缓存和分立缓存

分立缓存：指令和数据放在两个Cache中

4.3.2 Cache——主存地址映射

1.直接映射

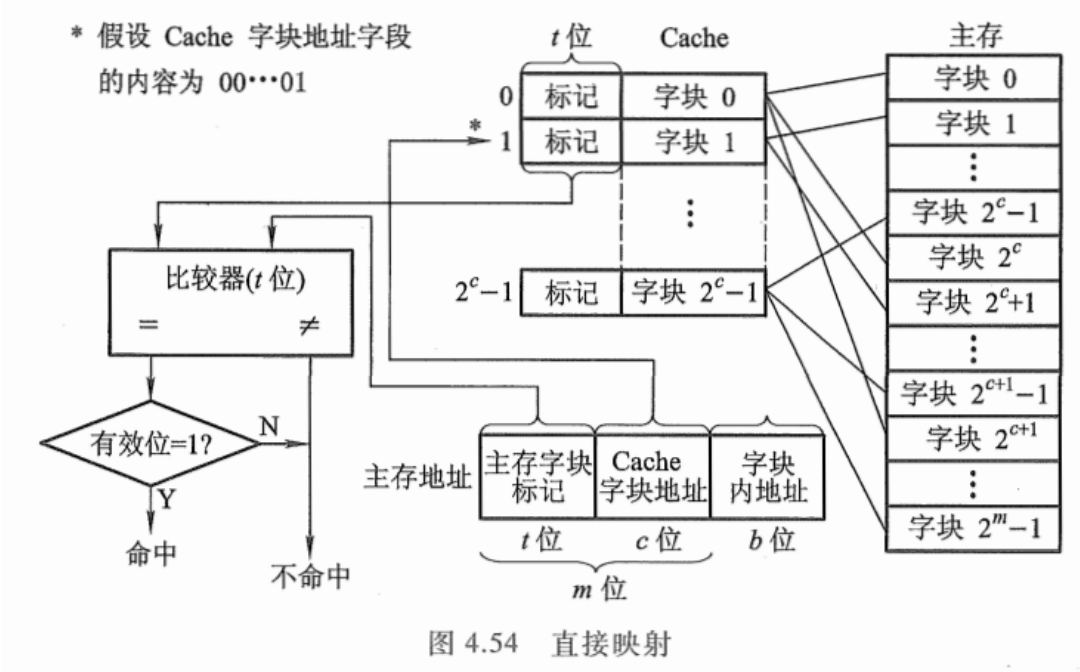


图 4.54 直接映射

每个主存块只与一个缓存块对应，映射关系式为

$i = jmodC$或$i = jmod2^C$

- i——缓存块号
- j——主存块号
- C——缓存块数

优点：实现简单，只需利用主存地址的某些位置就可以直接判断所需子块是否在缓存中

工作过程：

1. 当缓存收到CPU送来的主存地址后，只需根据中间c位字段，找到Cache字块
2. 判断字块的”标记“是否与主存地址的高t位相符

   若符合并且有效位为”1“（有效位用来识别Cache存储块中的数据是否有效），表示该Cache已和主存的某块建立了对应关系（即Cache命中），则可根据b位地址从Cache中获取信息。

   若不符合，或有效位为”0“，则从主存读入新的字块来代替旧的字块，同时将信息送往CPU，并修改Cache”标记“，如果原来有效位为”0“，则还需要将有效位改为”1“

2.全相联映射

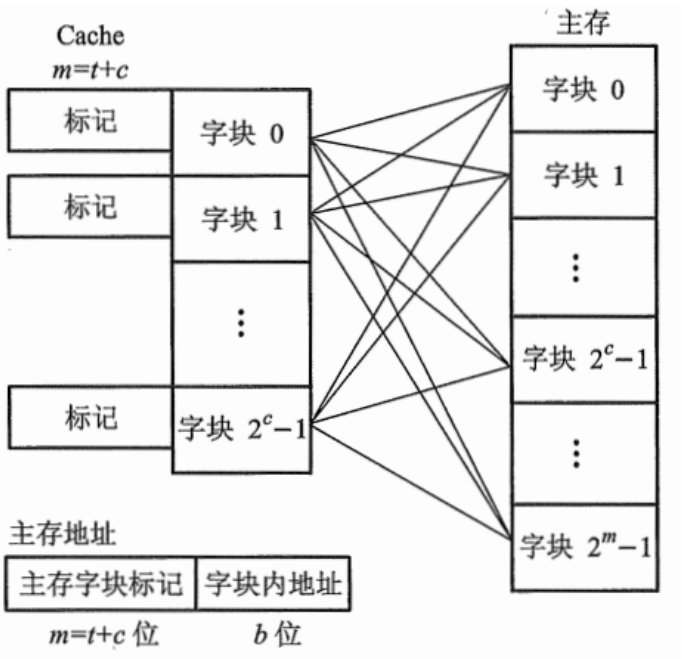


图 4.55　全相联映射

允许主存中的每一字映射到Cache中的任何一块位置上，这种映射方式可以从已被占满的Cache中替换出任一旧字块。

优点：这种方式灵活，命中率也更高，缩小了块冲突率

3.组相联映射

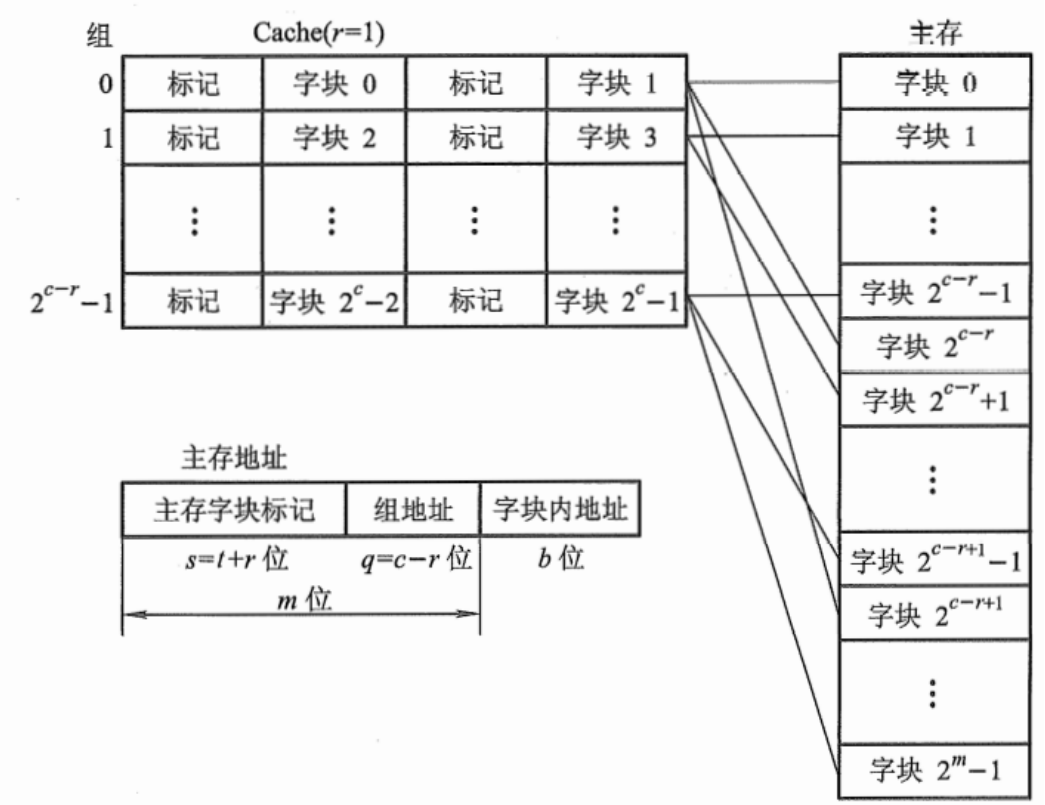


图 4.56　组相联映射

直接映射和全相联映射的一种折中。它把Cache分为Q组，每组有R块，并有以下关系：

$$i = j \ mod \ Q$$

组内2块的组相联映射叫做二路组相联映射

4.3.3 替换策略

当新的主存块需要调入Cache并且它的可用空间又被占满时，需要替换掉Cache的数据，这就产生了替换策略问题

在直接映射的Cache中，由于某一个主存字块只与一个Cache字块有映射关系，替换策略很简单

对于组相联映射和全相连映射中，替换策略有三种：

1. 先进先出(First-In-First-Out,FIFO)算法

选择最早调入Cache的字块进行替换，不需要记录各字块的使用情况，比较容易实现，开销小，但没有根据访存的局部性原理，故不能提高Cache命中率

2. 近期最少使用(Least Recently Used,LRU)算法

比较好地利用访存局部性原理，替换出近期使用得最少的字块。需要随时记录Cache中各字块的使用情况，以便确定哪个字块是近期最少使用的字块。它实际是一种推测的方法，比较复杂，一般采用简化的方法，只记录每一个块最近一次使用的时间。LRU算法的平均命中率比FIFO的高

3. 随机法

随机地确定被替换的块，比较简单，可采用一个随机数产生器产生一个随机地被替换的块，但它也没有根据访存的局部性原理，故不能提高Cache命中率。

4.4 辅助存储器

辅助存储器作为主存的后援设备又称为外部存储器，简称外存，它与主存一起组成了存储器系统的主存——辅存层次。
这一部分作为扩展内容，可根据情况自行阅读课本。

第5章 输入输出系统

5.1 概述

5.1.1 输入输出系统的发展概况

1. **早期阶段**

I/O设备通过CPU与主存交换信息

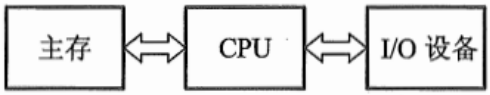


图 5.1　I/O 设备通过 CPU 与主存交换信息

特点：

- 每个I/O设备都必须配有一套独立的逻辑电路与CPU相连
- I/O设备与CPU串行工作
- 增添、撤减或更换I/O设备是非常困难的

2. **接口模块和DMA阶段**（本章重点）

这个阶段I/O设备通过接口模块与主机连接，计算机系统采用了总线结构

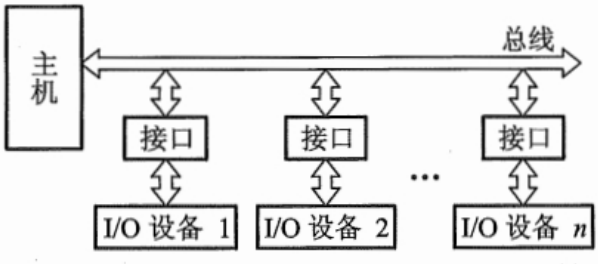


图 5.2　I/O 设备通过接口与主机交换信息

通常在接口中都设有数据通路和控制通路，数据经过接口既起缓冲作用，又可以完成串-并变换

CPU和I/O设备还不能做到绝对的并行工作

为了进一步提高工作效率，又出现了直接存储器存取技术(DMA技术)，其特点是I/O设备和主存之间有一条直接数据通路，I/O设备可以直接和主存交换信息

3. **具有通道结构的阶段**

在大型计算机中，I/O设备配置繁多，数据传送频繁，仍采用DMA方式会出现一系列的问题

- 每台I/O设备都需要配置专用的DMA接口，增加硬件成本，控制十分复杂
- CPU对众多DMA接口进行管理，降低CPU的整体工作效率

通道是用来管理I/O设备以及实现主存和I/O设备之间交换信息的部件，可视作一种具有特殊功能的处理器

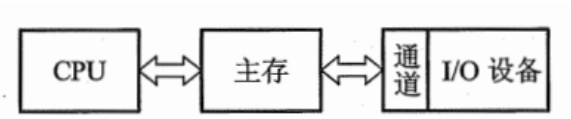


图 5.3　I/O 设备通过通道与主机交换信息

4. **具有I/O处理机的阶段**

I/O处理机又称为外围处理机，基本独立于主机工作，既可完成I/O通道要完成的I/O控制，又可以完成码制变换，格式处理，数据检错，纠错等操作

本章主要介绍第二阶段的输出输入系统

5.1.2 输入输出系统的组成

总体来说，输入输出系统由I/O硬件和I/O软件两部分组成

1. I/O软件

主要任务：

- 将用户编制的程序或数据输入主机内
- 将运算结果输送给用户
- 实现输入输出系统与主机工作的协调

(1)I/O指令

是机器指令的一类，其指令格式形如

$$\text{操作码} + \text{命令码} + \text{设备码}$$

操作码字段可作为I/O指令与其他指令(如访存、算逻、控制指令)的判别代码

命令码体现I/O设备的具体操作

设备码是多态I/O设备的选择码

(2)通道指令

通道指令是对具有通道的I/O设备专门设置的指令，一般用于指明参与传送的数据在主存中的首地址

2. I/O硬件

带接口的I/O系统中，包括接口模块和I/O设备

通道系统中，主要包括I/O设备、设备控制器和通道组成

5.1.3 I/O设备与主机的联系方式

1. I/O设备编码方式

通常将I/O设备码看作地址码，可采用两种方式编制

- 统一编制——将I/O地址看作存储器地址的一部分
- 不统一编制——不占用主存空间，但需设I/O专用指令

当设备通过接口与主机相连时，CPU可以通过接口地址访问I/O设备

2. 设备寻址

由于每台设备都被赋予一个设备号，因此，要启动某一设备时，可有I/O指令的设备码字段直接指出该设备的设备号。通过接口电路中的设备选择电路，便可选中要交换信息的设备

3. 传送方式

串行传送——同一瞬间只传输一位信息，传输速度较慢，但只需一根数据线和一根地线

并行传送——同一瞬间，n位数据同时从CPU输出至I/O设备，传送速度较快，要求数据线多

4. 联络方式

(1)立即访问方式

对于一些工作速度十分缓慢的I/O设备，如灯的开关，开关的通断，当它们与CPU发生联系时，通常都已使处于某种等待状态，因此只要CPU指令一到，它们便立即响应

(2)异步工作采用应答信号联络

CPU与I/O设备速度不匹配时使用

(3)同步工作采用同步时标联络

CPU与I/O设备速度匹配时使用

5.1.4 I/O设备与主机信息传送的控制方式

1. **程序查询方式**

由CPU通过程序不断查询I/O设备是否已做好准备，从而控制I/O设备与主机交换信息。

要求I/O接口内设置一个能反映I/O设备是否准备就绪的状态标记,CPU通过对此标记的检测，可得知I/O设备的准备情况

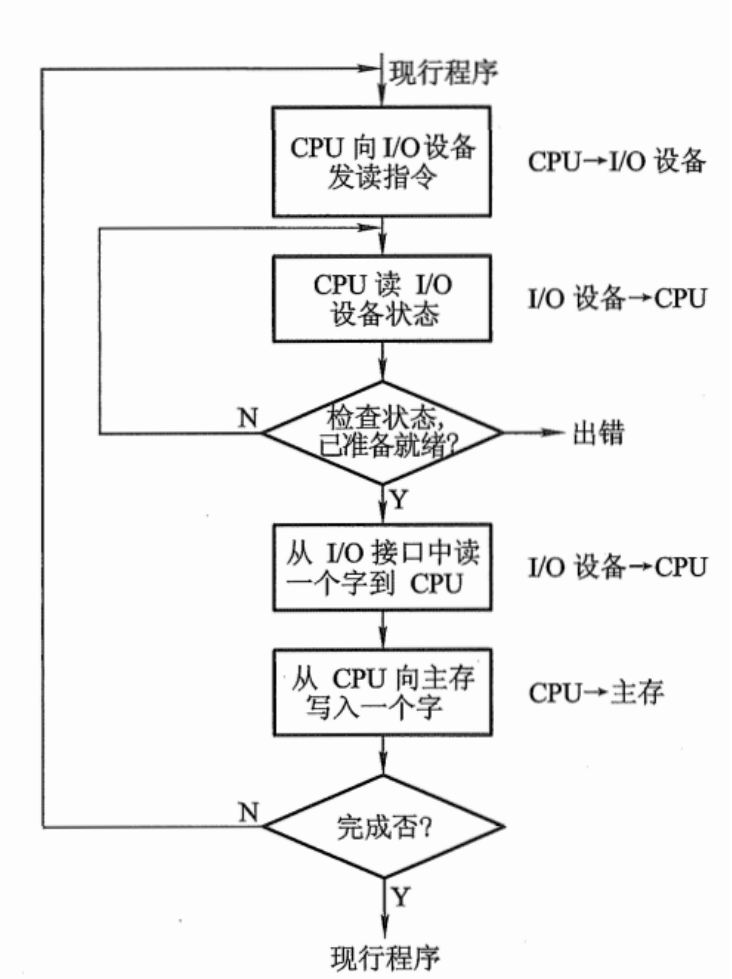


图 5.9　程序查询方式流程

CPU和I/O设备串行工作，CPU工作效率不高

2. **程序中断方式**

CPU启动I/O设备后，不查询设备是否以及准备就绪，继续执行自身程序，只是当I/O设备准备就绪并向CPU发出终端请求后才予以响应，这将大大提高CPU的工作效率

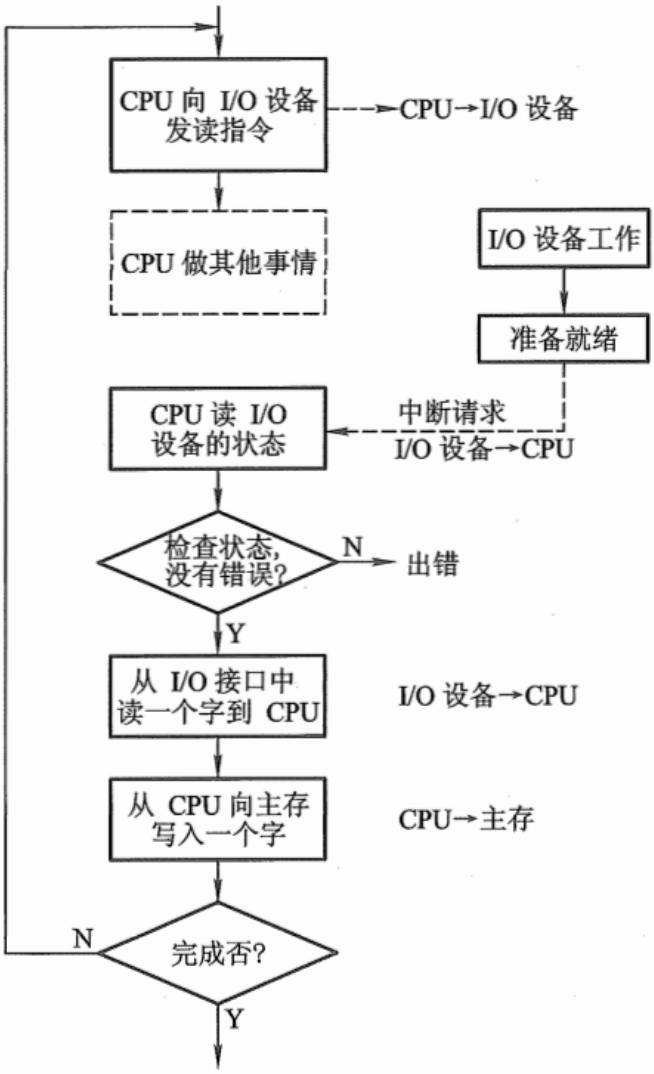


图 5.11　程序中断方式流程

CPU响应I/O设备的终端请求后，中断现行程序，转至终端服务程序，待处理完后又返回到原程序断点处，继续执行下一条指令

不会出现“踏步”现象，CPU资源得到充分的利用

3. **DMA方式**

主存与I/O设备之间有一条数据通路，主存与I/O设备交换信息时，无需调用终端服务程序。若出现DMA和CPU同时访问主存，CPU总是将总线占有权让给DMA，通常将DMA的这种占有称为窃取或者挪用，一般窃取一个存取周期，称作窃取周期或挪用周期，这时，CPU能继续做内部的操作

5.2 I/O设备

了解即可

5.3 I/O接口

5.3.1 概述

I/O接口通常是指主机与I/O设备之间设置的一个硬件电路以其相应的软件控制

5.3.2 接口的功能和组成

1. **总线连接方式的I/O接口电路**

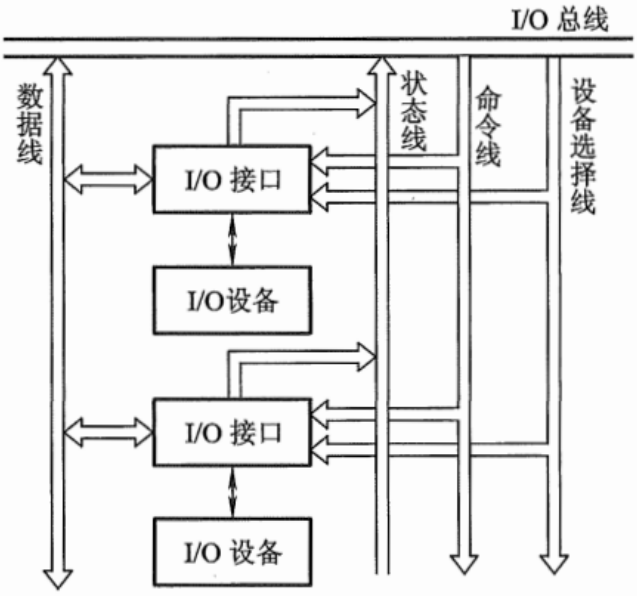


图 5.28　I/O 总线和接口部件

每一台I/O设备都是通过I/O接口挂到系统总线上的。图中的I/O总线包括数据线、设备选择线、命令线、状态线

(1)数据线

I/O设备与主机之间数据代码的传输线，根数一般与存储字长位数相同，通常是单向的

(2)设备选择线

用来传输设备码的，它的根数取决于I/O指令种设备码的位数

(3)命令线
命令线主要用以传输CPU向设备发出的各种命令信号。如启动、清除、屏蔽...

(4)状态线

将I/O设备的状态向主机报告的信号线

2. **接口的功能和组成**

功能:

- 选址功能

当设备选择线上的设备码与本设备设备码相同时，应发出设备选中信号SEL，这种功能可通过接口内的设备选择电路来实现

- 传送命令的功能

通常在I/O接口种设有存放命令的命令寄存器和命令译码器

命令寄存器用来存放I/O设备中的命令码，它受设备选中信号控制。命令线和所有接口电路的命令寄存器相连，只有被选中设备的SEL信号有效，命令寄存器才可接受命令线上的命令码

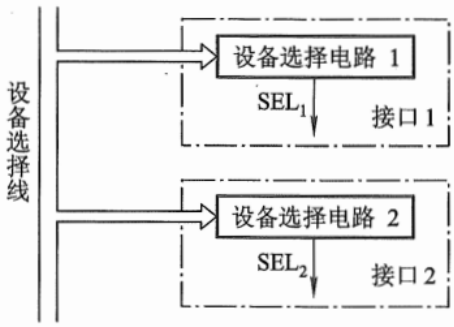


图 5.29　设备选择电路框图

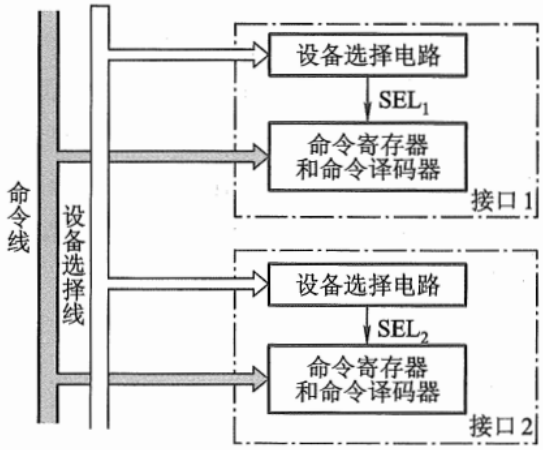


图 5.30　命令寄存器和命令译码器

- 传送数据的功能

接口中通常设有数据缓冲寄存器DBR，它用来暂存I/O设备与主机准备交换的信息，与I/O总线中的数据线相连

- 反映I/O设备工作状态的功能

完成触发器D，工作触发器B来标志设备所处的状态

5.3.3 接口的类型

1. 按数据传送方式分：

- 串行接口
- 并行接口

2. 按功能选择的灵活性：

- 可编程接口
- 不可编程接口

3. 通用性分类

- 通用接口
- 专用接口

4. 数据传送的控制方式分

- 程序型接口
- DMA型接口

5.4 程序查询方式

5.4.1 程序查询流程

核心问题在于每时每刻需不断查询I/O设备是否就绪

当I/O设备较多时，CPU需按照各个I/O设备在系统中的优先级别进行逐级查询

通常执行下面三条指令

1. 测试指令，查询I/O设备是否准备就绪
2. 传送指令，当I/O设备已准备就绪时，执行传送指令
3. 转移指令，若I/O设备未准备就绪时，执行转移指令，转至测试指令，继续测试I/O设备的状态

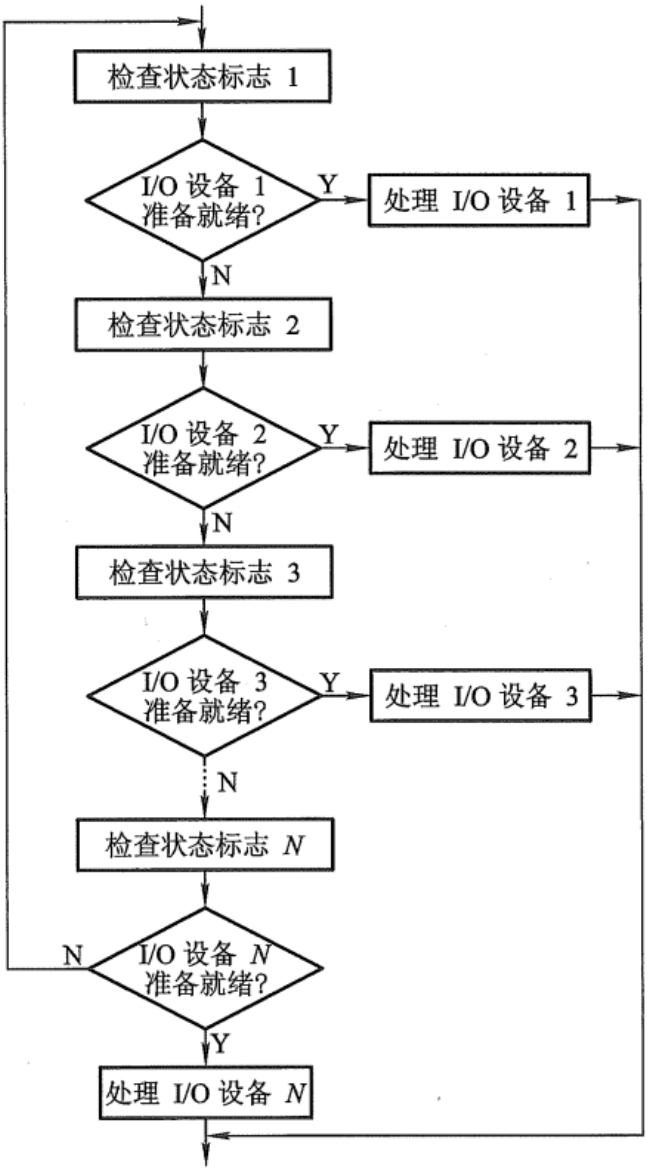
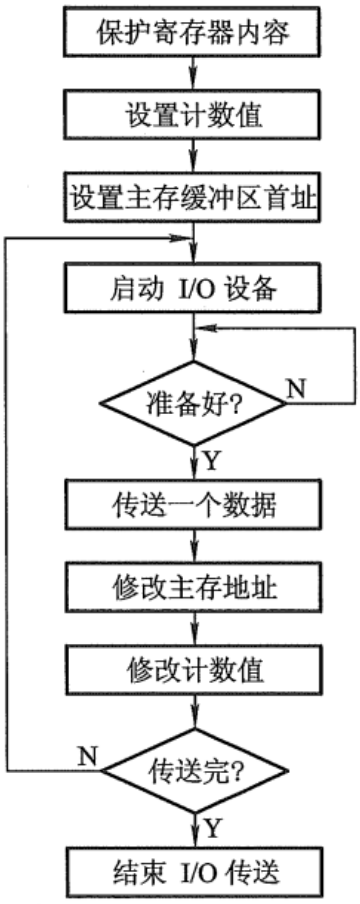


图 5.33　多个 I/O 设备的查询流程

程序流程框图



| 5.34　程序查询方式的程序流程

1.由于这种方式传送数据时要占用CPU中的寄存器，故首先需将寄存器原内容保护起来

2.传送的往往是一批数据，需要线设置I/O设备与主机交换数据的计数值

3.设置欲传送数据在主存缓冲区的首地址

4.CPU启动I/O设备

5.将I/O接口中的设备状态取至CPU并测试I/O设备是否准备就绪，如果未准备就绪，则等待至准备就绪为止。

6.CPU执行I/O指令，从I/O接口中的数据缓冲寄存器中取出数据或存入数据

7.修改主存地址

8.修改计数值

9.判断计数值若不为0，返回4

10.结束I/O传送

5.4.2 程序查询方式的接口电路

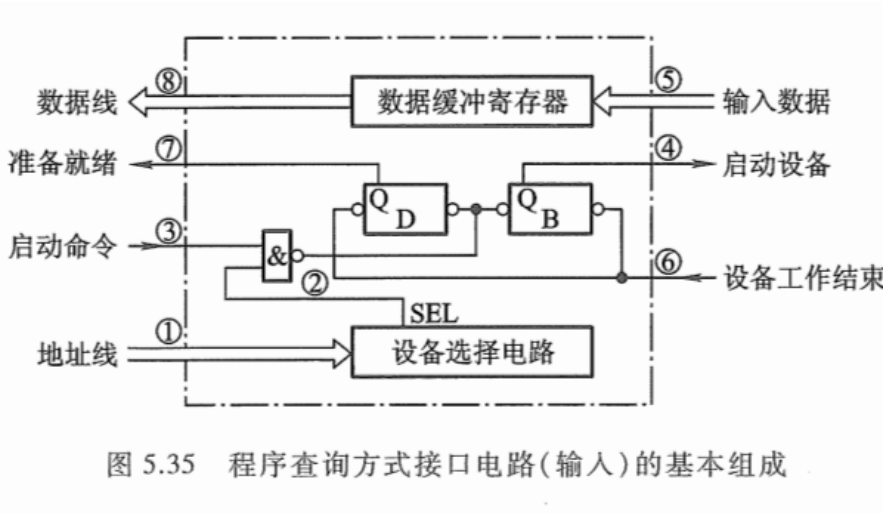


图 5.35　程序查询方式接口电路(输入)的基本组成

1. CPU通过I/O指令启动输入输出设备时，该设备的设备码字段通过地址线送至设备选择电路
2. 如果该接口上的设备码与地址线上的代码吻合，其输出SEL有效
3. I/O设备的启动命令经过“与非”门将工作触发器B置"1",完成触发器置"0"
4. 由B触发器启动I/O设备
5. 输入设备将数据送至数据缓冲寄存器
6. 设备工作结束，将触发器B置“0”，触发器D置“1”，表示外设准备就绪
7. D触发器将准备就绪的状态送至CPU，表示“数据缓冲满”
8. CPU执行输入指令，将数据缓冲寄存器中的数据送至CPU的通用寄存器，再存入主存相应单元

5.5 程序中断方式

5.5.1 中断的概念

计算机再执行程序的过程中，当出现异常情况或特殊请求时，计算机停止现行程序的运行，转向对这些异常情况或特殊请求的处理，处理结束后再返回到现行程序的间断处，继续执行原程序

5.5.2 I/O中断的产生

在I/O设备与主机交换信息时，由于设备本身机电特性的影响，其工作速度较低，与CPU无法匹配，因此，CPU启动设备后，往往需要等待一段时间才能实现主机与I/O设备之间的信息交换。如果在设备准备的同时，CPU不做无谓的等待，而继续执行现行程序，只有当I/O设备准备就绪向CPU提出请求后，再暂时中断CPU现行程序转入I/O服务程序，这便产生了I/O中断

5.5.3 程序中断方式的接口电路

1. **中断请求触发器和中断屏蔽触发器**

每台外部设备需要配置一个中断请求触发器INTR，当其置"1"时，表示该设备向CPU提出中断请求。但该设备欲提出中断请求时，其设备本身必须准备就绪，即接口内的完成触发器D的状态必须为"1"

当多个中断源同时提出请求时，CPU必须对各中断源的请求进行排队，且只能接受级别最高的中断源的请求，不允许级别低的中断源中断正在运行的中断服务程序。这样，在I/O接口中需设置一个屏蔽触发器MASK，当其为"1"时，表示被屏蔽，即封锁其中断源的请求。

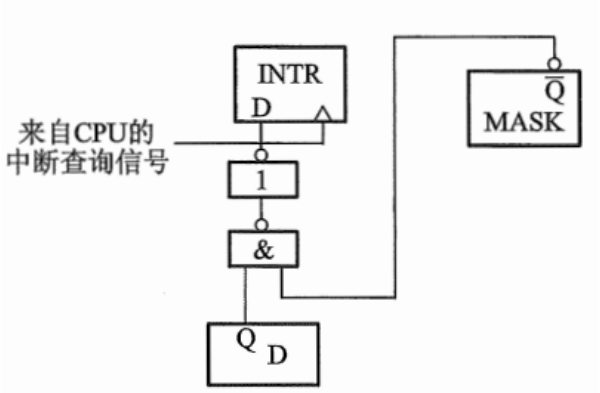


图 5.37　接口电路中 D、INTR、MASK 和中断查询信号的关系

2. **排队器**

设备的优先级处理可以采用硬件方法，也可采用软件方法(第八章中会介绍)

硬件排队器的实现方法有很多，可以在CPU内设置一个统一的排队器，对所有的中断源进行排队，也可以在接口电路内分别设置各个设备的排队器，下面给出设在各个接口电路中的排队器电路，又称链式排队器。

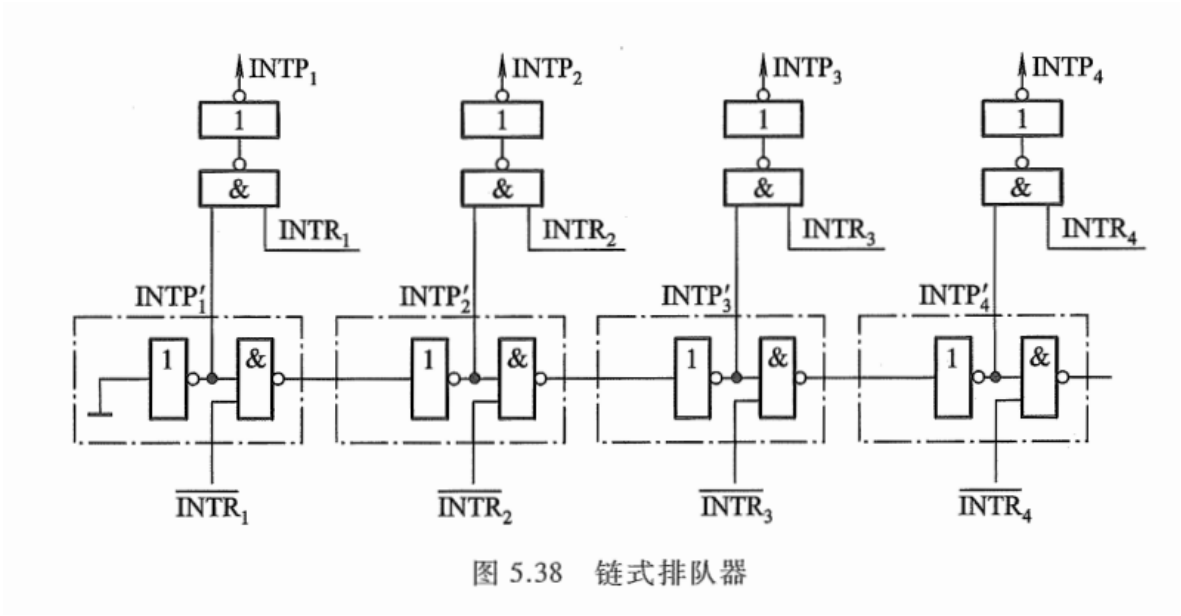


图 5.38　链式排队器

3. **中断向量地址形成部件(设备编码器)**

CPU一旦响应了I/O中断，就要暂停现行程序，转去执行该设备的中断服务程序

入口的寻找方式也可以用硬件和软件的方式来完成

这里介绍硬件向量法，就是通过向量地址来寻找设备的中断服务程序入口地址，而且向量地址是由硬件电路产生的

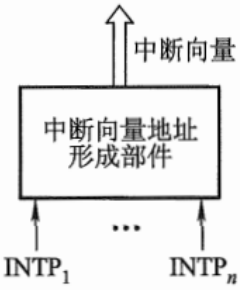


图 5.39　中断向量地址形成部件框图

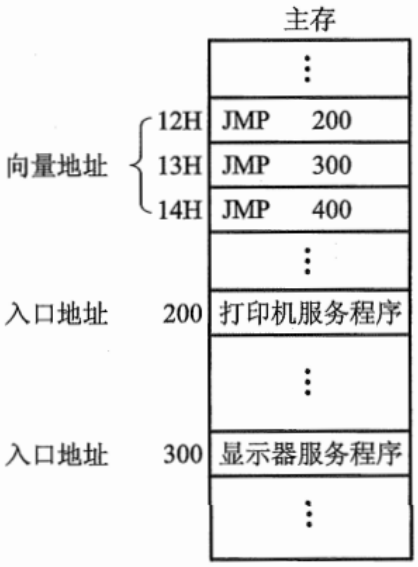


图 5.40　通过向量地址寻找入口地址

5.5.4 I/O中断处理过程

1. **程序中断方式接口电路的基本组成**

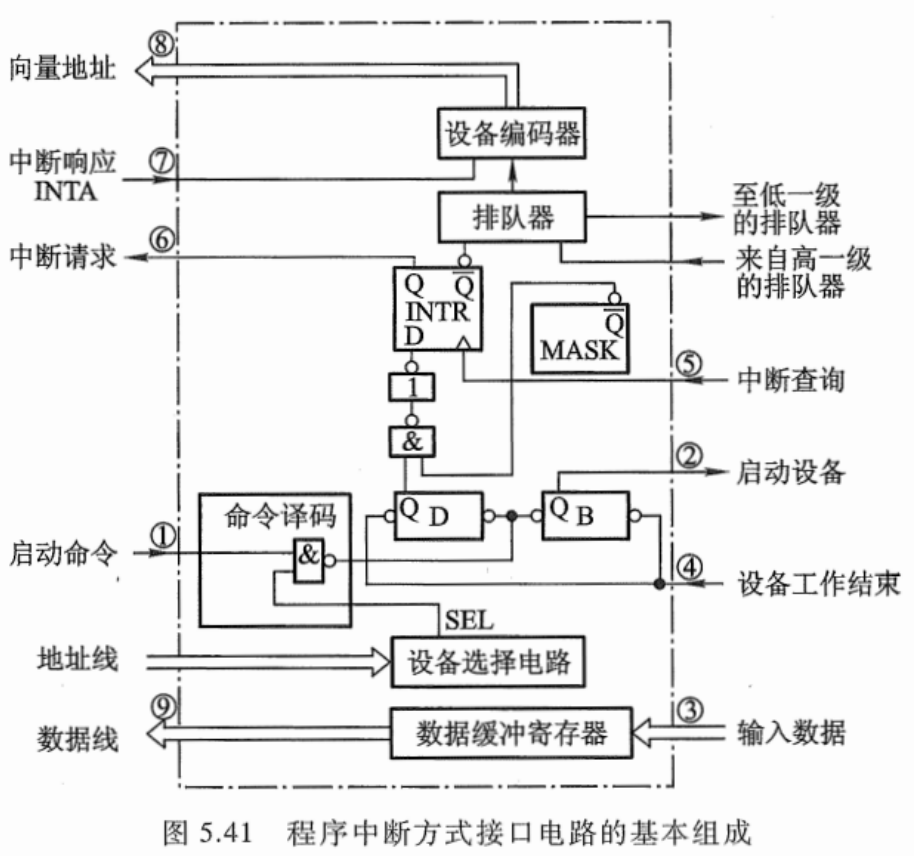


图 5.41　程序中断方式接口电路的基本组成

2. **CPU响应中断的条件和时间**

条件：CPU中的允许中断触发器EINT为"1"

该触发器可用开中断指令置位，也可用关中断指令或硬件自动使其复位

CPU是在统一的时刻(每条指令执行阶段结束前)向接口发中断查询信号，以获取I/O的中断请求。

CPU响应中断的时间一定是每条指令执行阶段的结束时刻

3. I/O中断处理过程

当CPU通过I/O指令选中每台I/O设备后：

1. 由CPU发启动I/O设备命令，将接口中的B置“1”，D置“0”
2. 接口启动输入设备开始工作
3. 输入设备将数据送入数据缓冲寄存器
4. 输入设备向接口发送“设备工作结束”信号，将B置"0"，D置“1”，标志设备准备就绪
5. 当设备准备就绪，且本设备未被屏蔽(MASK=0),在指令执行阶段的结束时刻，由CPU发出**中断查询**信号
6. 设备中断请求INTR被置“1”，标志设备向CPU提出**中断请求。**与此同时，INTR被送至排队器，进行**中断判优。**
7. 若CPU允许中断(EINT=1)，设备又被排队选中，即进入**中断响应**阶段，由中断响应信号INTA将排队器输出送至编码器形成向量地址
8. 向量地址送至PC,作为下一条指令的地址
9. 由于向量地址中存放的是一条无条件转移指令，故这条指令执行结束后，无条件转至该设备的服务程序入口地址，开始执行中断服务程序，进入**中断服务**阶段，通过输入指令将数据缓冲寄存器的输入数据送至CPU的通用寄存器，再存入主存相关单元
10. 中断服务程序的最后一条指令是中断返回指令，当其执行结束时，即**中断返回**至原程序断点处。

$$\text{中断请求} -> \text{中断判优} -> \text{中断响应} -> \text{中断服务} -> \text{中断返回}$$

5.5.5 中断服务程序的流程

$$\text{保护现场} -> \text{中断服务} -> \text{恢复现场} -> \text{中断返回}$$

1. **保护现场**

- 保存程序的断点，由中断隐指令完成
- 保存通用寄存器和状态寄存器的内容，由中断服务程序完成

2. **终端服务**

中断服务程序的主体部分，对于不同的中断请求源，其中断服务操作内容是不同的

3. **恢复现场**

退出服务程序前，将原程序中断时的“现场”恢复到原来的寄存器中

4. **中断返回**

中断服务程序的最后一条指令通常是一条中断返回指令，使其返回到原程序的断点处，以便执行原程序

计算机在处理中断的过程中，有可能会出现新的中断请求，如果CPU暂停现行的中断服务程序，转去处理新的中断请求，这种现象叫做中断嵌套或多重中断。如果CPU不理睬新的中断请求，这种中断叫做单重中断。

这两种中断的区别在于开中断设置时间不同

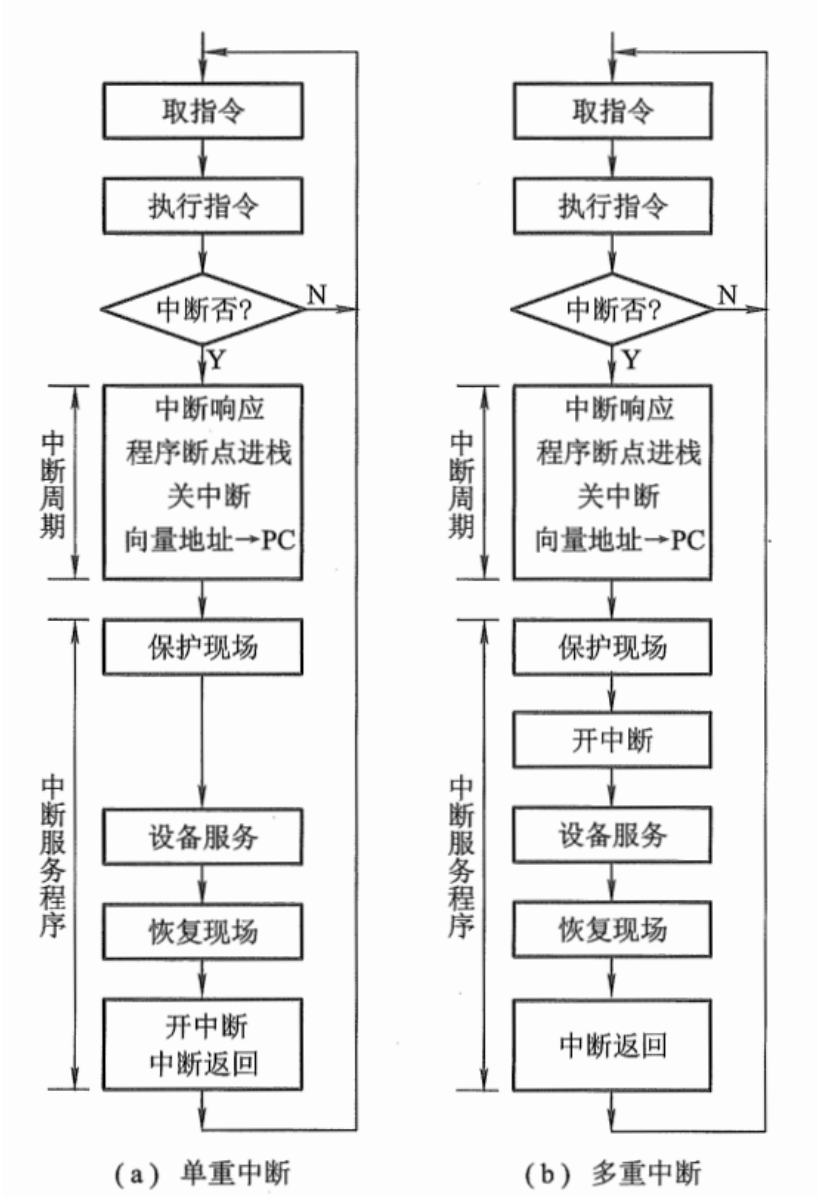


图 5.43 单重中断和多重中断服务程序流程

程序中断方式实现了CPU和I/O设备的并行工作，提高了CPU的资源利用率。从微观角度分析，CPU字处理中断服务程序时仍需要暂停原程序的正常运行

5.6 DMA方式

5.6.1 DMA方式的特征

由于主存和DMA接口之间有一条数据通路，因此主存和设备交换信息时，不通过CPU，也不需要CPU暂停现行程序为设备服务，省去了保护现场和恢复现场，因此工作速度比程序中断方式的工作速度高

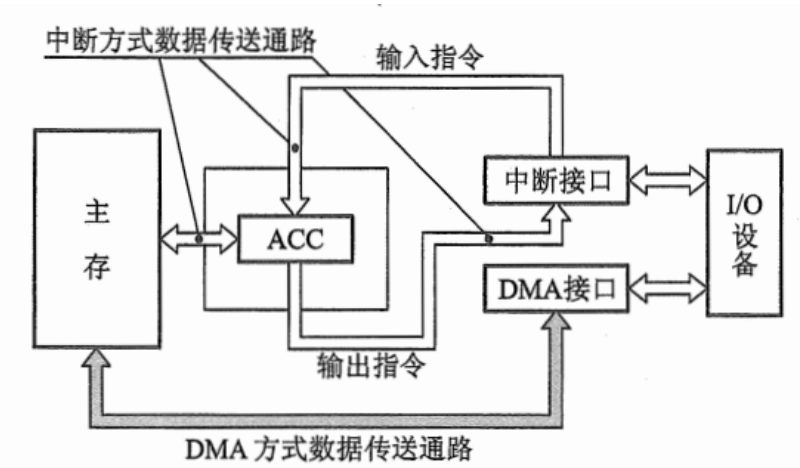


图 5.45　DMA 和程序中断两种方式的数据通路

在DMA方式中，由于DMA接口与CPU共享主存，这就可能出现两者争用内存的冲突。为了有效地分时使用主存，通常DMA与主存交换数据时采用如下三方法：

1. **停止CPU访问主存**

但外设要求传送一批数据时，由DMA接口向CPU发一个停止信号，要求CPU放弃地址线，数据线和有关控制线地使用权。DMA接口获得总线控制权后，开始进行数据传送，在数据传送结束后，DMA接口通知CPU可以使用主存，并把总线控制权交回给CPU

优点：控制简单，适用于数据传输率很高的I/O设备实现称组数据的传送

缺点：DMA接口在访问主存时，CPU基本上处于不工作或保持原状态。CPU对主存的利用率没有得到充分发挥

2. **周期挪用**

每当I/O设备发出DMA请求时，I/O设备便挪用或窃取总线占用权一个或几个主存周期，而DMA不请求时，CPU仍继续访问主存

- CPU不需要访问主存——I/O设备与CPU不发送冲突
- CPU正在访问主存——待存取周期结束，CPU才将总线占有权让出
- CPU要求访问主存时，I/O设备也要求访问主存——I/O访存优先于CPU访存，因为I/O不立即访问主存可能会导致数据丢失

3. DMA与CPU交替访问

适合于CPU的工作周期比主存存取周期长的情况

5.6.2 DMA接口的功能和组成

1. **DMA接口的功能**

- 向CPU申请DMA传送
- 在CPU允许DMA工作时，处理总线控制权的转交，避免因进入DMA工作而影响CPU正常活动或引起总线竞争
- 在DMA期间管理系统总线，控制数据传送
- 确定数据传送的起始地址和数据长度，修正数据传送过程中的数据地址
- 在数据块传送结束时，给出DMA操作完成的信号

2. **DMA接口基本组成**

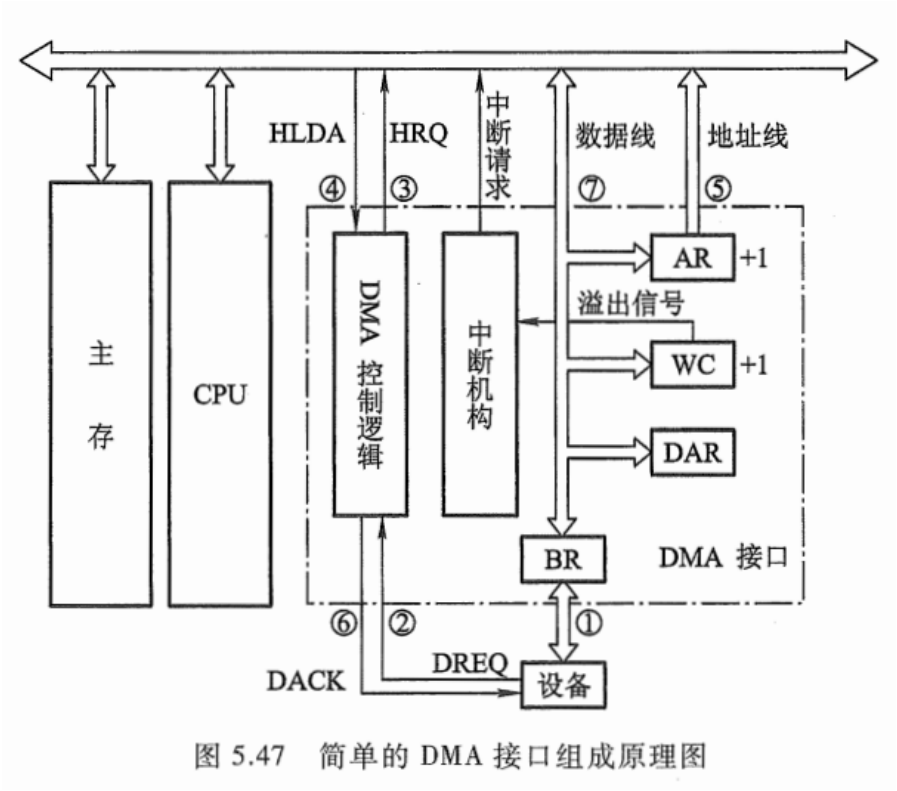


图 5.47　简单的 DMA 接口组成原理图

(1)主存地址寄存器(AR)

AR用于存放主存中需要交换数据的地址。在DMA传送数据前，必须通过程序将数据在主存中的首地址送到主存地址寄存器。在DMA传送过程中，每交换一次数据，将地址寄存器内容加1，直到传送完成为止

(2)字计数器(WC)

用于记录传送数据的总字数，通常以交换字数的补码值预置，传送一个就加1

(3)数据缓冲寄存器(BR)

用于暂存每次传送的数据，可能还包括装配或拆卸字信息的硬件逻辑

(4)DMA控制逻辑

负责管理DMA的传送过程

(5)中断机构

当字计数器溢出时，表示一批数据交换完毕，由“溢出信号”通过中断机构向CPU提出中断请求，请求CPU作DMA操作的后处理。 与上述的中断不是一种操作

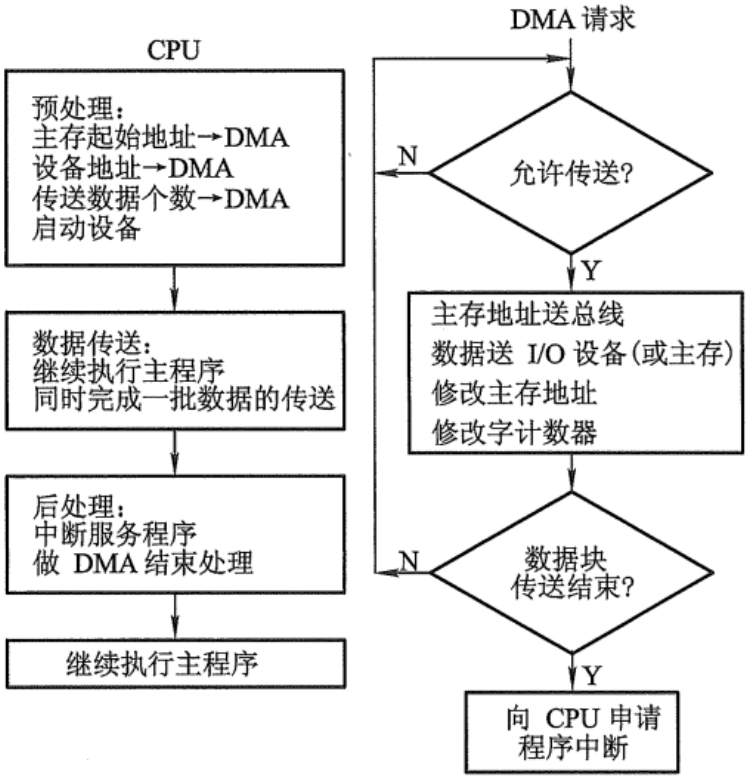
(6)设备地址寄存器(DAR)

存放I/O设备的地址码或表示设备信息存储区的寻址信息

5.6.3 DMA的工作过程

1. DMA传送过程

预处理— ＞ 数据传送— ＞ 后处理



（a） DMA 传送过程　　（b） 数据传送阶段的细化

图 5.48　DMA 传送过程示意图

（1）**预处理**

在DMA接口开始工作之前，CPU必须给它预置以下信息

- 给DMA控制逻辑指明数据传输的方向是输入(写内存)还是输出(读内存)
- 给DMA设备地址寄存器送入设备号，并启动设备
- 向DMA主存地址寄存器送入交换数据的主存起始地址
- 对字计数器赋予交换数据的个数

上述工作由CPU执行几条输入输出指令完成，即程序的初始化。这些工作完成后，CPU继续执行原来的程序。

当I/O设备准备号发送的数据或上次接收的数据已经处理完毕时，它便通过DMA接口向CPU提出占用总线的申请，多个DMA同时申请，就按照轻重缓急由硬件排队判优逻辑决定优先。待I/O设备得到主存控制权后，数据的传送便由该DMA接口进行管理

(2)数据传送

DMA方式是以数据块为单位传送的，以周期挪用的DMA方式为例

其数据输入的具体操作如下：

- 当设备准备号一个字，发出选通信号，将该字读到DMA的数据缓冲寄存器内，表示数据缓冲寄存器满
- 与此同时设备向DMA设备发送请求DREQ
- DMA接口向CPU申请总线控制权HRQ
- CPU发回HLDA信号，表示允许将总线控制权交给DMA接口
- 将DMA主存地址寄存器中的主存地址送地址总线，并命令存储器写
- 通知设备已被授予一个DMA周期DACK，并为交换下一个字做准备
- 将DMA数据缓冲寄存器的内容送至地址总线指定的单元中
- 修改主存地址和字计数值
- 判断数据块是否传送结束，若未结束继续传送，结束，则向CPU申请程序中断，标志数据块传送结束。

其数据输出的具体操作如下：

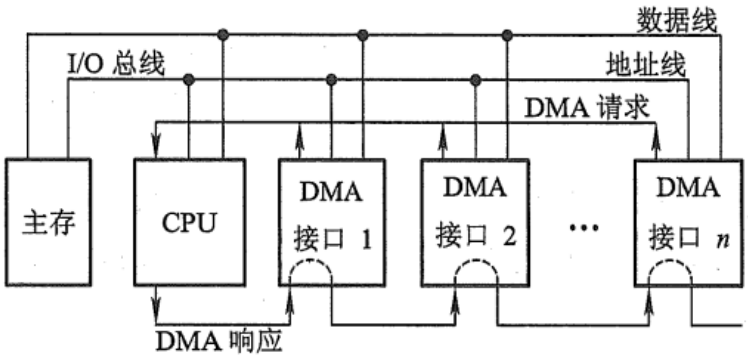
- 当DMA数据缓冲寄存器已将输出数据送至I/O设备后，表示数据缓冲寄存器已空
- 设备向DMA设备发送请求DREQ
- DMA接口向CPU申请总线控制权HRQ
- CPU发回HLDA信号，表示允许将总线控制权交给DMA接口
- 将DMA主存地址寄存器中的主存地址送地址总线，并命令存储器读
- 通知设备已被授予一个DMA周期DACK，并为交换下一个字做准备
- 主存将响应地址单元的内容通过数据总线读入DMA的数据缓冲寄存器中
- 将DMA数据缓冲寄存器的内容送到输出设备，若为字符型设备，则需将其拆称字符输出
- 修改主存地址和字计数值
- 判断数据块是否传送结束，若未结束继续传送，结束，则向CPU申请程序中断，标志数据块传送结束。

(3)后处理

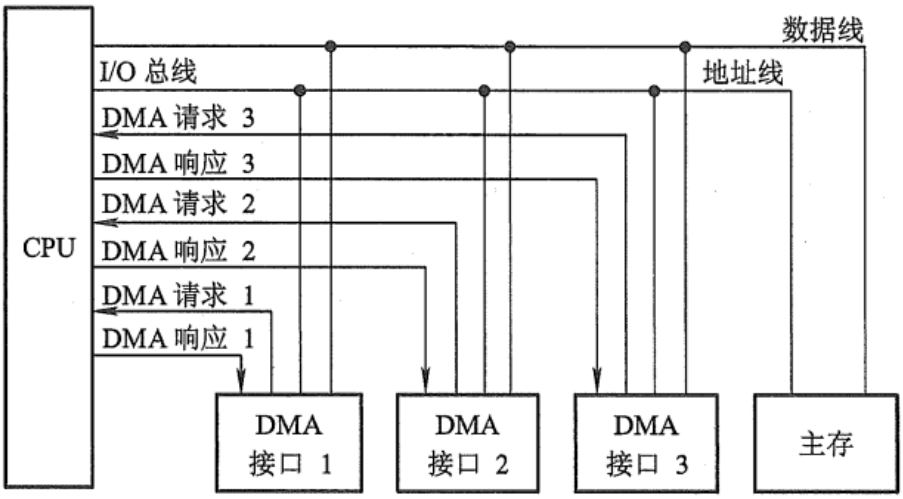
当DMA的中断请求得到响应后，CPU停止原程序的执行，转去执行中断服务程序，做一些DMA的结束工作。包括校验送入主存的数据是否正确，决定是否继续使用DMA传送其他数据块等操作

2. DMA接口与系统的连接方式

有公共请求线的和独立的DMA请求方式



（a） 具有公共请求线的 DMA 请求



（b） 独立的 DMA 请求

图 5.49　DMA 接口与系统的连接方式

3. DMA小结

与程序中断方式相比，DMA方式有以下特点：

- 从数据传送看，程序中断方式靠程序传送，DMA方式靠硬件传送
- 从CPU响应时间看，程序中断方式是在一条指令执行结束时响应，DMA方式可在指令周期内的任意存取周期结束时响应
- 程序中断方式有处理异常事件的能力，DMA方式没有这种能力
- 程序中断方式需要中断执行程序，故需保护现场，DMA方式不中断现行程序，无需保护现场
- DMA的优先级要比程序中断方式优先级高

5.6.4 DMA接口的类型

1. 选择型DMA接口 物理上允许连接多个设备，逻辑上只允许一个设备工作
2. 多路型DMA接口 物理和逻辑上都允许多个设备同时工作

5.7 练习题

1. I/O设备与主机交换信息时，共有哪几种控制方式。简述它们的特点
2. 以键盘设备为例，结合中断接口电路，说明其工作过程
3. 是比较单重中断和多重中断服务程序的处理流程，说明它们不同的原因
4. 结合DMA接口电路说明其工作过程