计算机组成原理

- ---

第二章<数据的表示和运算>
  - 编码
    - BCD编码： 23 -> 0010 0011
    - 海明码校验：$2^k >= n + k + 1$
      能发现2位错误，纠正一位
    - 循环冗余码CRC校验：模2除法，能发现并纠正一位or多位错误
  - 定点数的表示和计算
    - 原码 +0：0 0000 -0：1 0000
    - 补码：正=原 负=符号不变取反加一 ±0=0.0000
    - 反码：正=原 负=符号不变取反不加一 +0=0.1111 -0=1.0000
    - 移码：先求补码，将其符号位取反（为了直接看出数之间的大小关系）
    - 进进位
    - 符号扩展：补码左0右1
    - 溢出的判断 单符号位 双符号位..00 11表示无溢出 01正溢出 10负溢出
    - 定点数的 乘法(1.原码一位乘法 2.补码一位乘法) 除法(1.原码除法运算[不恢复余数法] 2.补码除法运算[加减交替法])
    - 强制类型转换 int->float->double
  - 浮点数的表示和运算
    - 阶码（常用补码或移码表示）尾码（常用原码或补码表示）
    - 规格化浮点数：为提升运算的精度，规定尾数最高位必须是一个有效位
    - 阶码的加法操作：1>对阶 2>尾数求和 3>规格化 4>舍入:"0舍1入" 5>溢出判断由阶码的符号位决定!
- ---

第三章<存储系统>
  - **层次结构**
    - 主存：CPU可以直接进行访问的，也可以和高速缓存器以及辅存交换数据
    - 辅存：存放暂时不用的数据
    - 高速缓冲存储器：位于CPU和主存之间，用来存放正在执行的程序段和数据
    - 按介质分类
      - 随机存储器RAM：主要用于主存和高速缓存
      - 只读存储器ROM：只能随机读，但不能随机写入。断电不会丢
      - 串行访问存储器：需按其物理地址顺序寻址，包括顺序存储器（磁带）与直接存取存储器（磁盘）
    - 按信息的可保存性分类
      - 易失性存储器：RAM
      - 非易失性存储器：ROM 磁表面存储器和光存储器
      - 破坏性读出和非破坏性读出（破坏性：每次读出后，要立即接一个再生操作，以便恢复被破坏的数据）
    - 存储器的性能指标
      - 存储容量：存储字节*字长
      - 单位成本：每位价格=总成本/容量
      - 存储速度：数据传输率=数据的宽度/存储周期
    - 存取时间：存取时间时指从启动一次存储器操作到完成该操作所经历的时间，分为读出时间和写入时间。
    - 存取周期： 它是指存储器进行一次完整的读写操作所需的全部时间，即连续两次独立访问存储器操作（读或写操作）之间所需的最小时间间隔。
    - 主存带宽：主存带宽又称数据传输率，表示每秒从主存进出信息的最大数量，单位为字/秒，字节/秒存取时间不等于存储周期，通常存储周期大于存取时间。因为任何一种存储器，在读写操作之后，总要有一段恢复内部状态的复原时间。
  - **层次化结构**
    - CPU<-Cache<-主存<-磁盘<-磁带，光盘
    - cache-主存：解除速度不匹配问题 主存-辅存：解决存储系统容量问题
    - cache-主存，主存-辅存之间的数据调动都是由硬件和操作系统完成的，对程序员是透明的
  - **半导体随机存储器**
    - 地址线：单向输入的，位数与存储字的个数有关
    - 片选线：选择用哪一片信号
    - 数据线：双向的，位数与读出或写入的数据位数有关，如地址线10根，数据线8根，芯片的容量就是 $2^{10} * 8 = 8K$
    - SRAM（静态随机存储器）：存储元是用**双稳态触发器**来记忆信息的，具有非破坏性读出。但是易失性半导体存储器
    - DRAM（动态随机存储器）：利用存储元电路中栅级电容上的电荷来存储信息的，常见的DRAM的基本存储电路通常分为三管式和单管式。DRAM采用地址复用技术，地址线是原来的1/2，且地址信号分行，列两次传送。但DRAM电容上的电荷一般只能维持1~2ms，因此即使电源不掉电，信息也会自动消失。所以要每隔一段时间刷新一次。
      - 1> 集中刷新，在一个刷新周期中，利用一段固定的时间进行刷新。（存在死区）
      - 2> 分散刷新，对一个刷新周期，前半部分正常读，写或保持；后半部分刷新某一行。（没有死区）
      - 3> 异步刷新，将上面两种结合。（有死区） DRAM要求，至少**2ms**更新所有行一次，计算刷新周期
    - 只读存储器（ROM）ROM
      - 1> 结构简单，所以位密度比可读写存储器的高
      - 2> 具有非易失性，所以可靠性高
    - ROM的类型
    - 1> 掩模式只读存储器 2> 一次可编程只读存储器 3> 可擦除可编程只读存储器 4> 闪速存储器 5> 固态硬盘
  - **主存储器与CPU的连接**
    - 主存通过 数据总线 地址总线 控制总线 与CPU连接
    - 数据总线的位数与工作频率的乘积正比于数据传输率
    - 地址总线的位数决定了可寻址的最大内存空间
    - 控制总线（读/写）指出总线周期的类型和本次输入/输出操作完成的时刻
    - 容量扩展
      - 位扩展： 将多个存储芯片的地址端，片选端和读写控制端相应并联
      - 字扩展：由片选信号识别芯片地址范围
    - 片选一般有两种方法线选法和译码片选法
      - 线选法：高地址直接分别接至各个存储芯片的片选端
      - 译码片选法： 高位地址线通过地址译码器芯片产生片选信号。
  - **双端口RAM和多模块存储器**
    - 双端口RAM： 有左，右两个独立的端口，两组相互独立的地址线，数据线和读写控制线，允许两个独立的控制器同时异步地访问存储单元。
      - 同时写会有写入错误 一读一写会有读出错误 通过设置忙信号 “0” 解决
    - 多模块存储器
      - 单体多字存储器：只有一个存储体，每个存储单元存储m个字，总线宽度也为m个字。一次并行读出m个字，地址必须顺序排列并处于同一存储单元
        - 缺点：指令和数据在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
      - 多体低位交叉存储器

        - 低位交叉编址：高位地址表示体号，低位地址为体内地址
        - 低位交叉编址： 采用流水线的方式并行存取，提高存储器的带宽
    - **高速缓冲存储器**
        - 程序访问的局部性原理包括时间局部性和空间局部性
        - Cache常由SRAM构成
            - 直接映射： 模2运算 块冲突，原来的块将无条件地被替换出去
            - 全相联映射： 把主存数据块装入Cache中的任何位置(随便映射)
            - 组相联映射： 将Cache空间分成大小相同的组，主存的一个数据块可以装入一组内的任何一个位置，即组间采取之间映射，组内采用全相联映射
            - 替换算法：随机（RAND） 先进先出（FIFO） 最近最少使用（LRU） 最不经常使用（LFU）
            - 写策略 全写法：当CPU对Cache写命中时，必须把数据同时写入Cache和主存
                写回法：当CPU对Cache写命中时，只修改Cache的内容，而不立即写入主存，只有当此块被换出时才写回主存
                写分配法：加载主存中的块到Cache中，然后更新这个Cache块
                非写分配法：只写入主存，不进行调块
                非写分配法通常与全写法合用，写分配法通常和写回法合用
    - **虚拟存储器**：CPU使用虚地址时，由辅助硬件找出虚地址和实地址之间的对应关系 实地址=主存页号+页内字地址 虚地址=虚存页号+页内字地址
        - 页式： 虚拟空间与主存空间都被划分成同样大小的页， 虚地址到实地址的变化由页表来实现； 装入位为“1”，则表示该页面已在主存中, 装入位为“0”，表示该页面不在主存中，此时要启动I/O系统，把该页从辅存调入主存后再供CPU使用。
        - 段式： 按程序的逻辑结构划分，各个段的长度因程序而定。 把虚地址分为两部分：段号和段内地址
        - 段页式： 每一个程序对应一个段表，每段对应一个页表，段的长度必须是页长的整数倍，段的起点必须是某一页的起点。以页为基本的传送单位。
        - 快表TLB： 根据程序执行的局部性原理，将经常访问的某些页对应的页项放入高速缓冲器组成快表，相应的把存放在主存中的页表称为慢表（Page）
-

第四章<指令系统>

- **指令格式** **操作码字段和地址码字段**
    - 指令（又称机器指令）是指示计算机执行某种操作的命令，是计算机运行的最小功能单位。 计算机的所有指令的集合构成该机的指令系统，也称为指令集合。
    - 指令的长度：取决于操作码的长度，操作数地址码的长度和操作数地址的个数
    - 根据操作数地址码的数目的不同，分类
        - 零地址指令：<OP> 不需要操作数，如：停机指令，关中断指令
        - 一地址指令：<OP|A1>
        - 二地址指令：<OP|A1（目的操作数）|A2（源操作数）> **目的操作数地址还用于保存此次运算的结果**
        - 三地址指令：<OP|A1（目的操作数）|A2（源操作数）|A3（结果）> 例：(A1)OP(A2)-->A3，需要访问4次主存
        - 四地址指令：<OP|A1（目的操作数）|A2（源操作数）|A3（结果）|A4（下址）>
    - 定长操作码指令格式，是指在最高位部分分配固定的若干位（定长）表示操作码。当计算机字长为32位或更长时，这是常规方法。
    - 扩展操作码指令，操作码的长度随地址码的减少而增加，不同地址数的指令可具有不同长度的操作码，从而在满足需要的前提下，有效地缩短指令字长。
    - 一般情况下，对使用频率较高的指令分配短操作码，对使用频率较低的指令分配较长的操作码
- **寻址方式**
    - 指令寻址： 寻找下一套将要执行的指令
        - 顺序寻址：通过PC+1,自动形成下一条指令
        - 跳跃寻址：本条指令给出的下条指令地址 ，跳跃结果是当前指令修改PC值，所以下一条指令仍然通过程序计数器PC给出
    - 数据寻址： 寻找指令中表示的操作数或怎样计算出操作数的地址
        - 隐含寻址： 指令不明显给出操作数的地址，不在地址字段中指出第二操作数，而规定**累加器ACC**作为第二操作数地址，累加器相对于单地址指令就是隐含指令
        - 立即寻址：地址字段就是操作数本身
        - 直接寻址：指令中的形式地址A是操作数的真实地址（EA)， EA= A
        - 间接寻址： 指令的地址字段给出的形式地址是操作数有效地址所在的存储单元的地址（访存3次）
        - 寄存器寻址： 指令字中直接给出操作数所在的寄存器编号，EA=Ri
        - 寄存器间接寻址：寄存器R中给出的不是一个操作数，而是操作数所在主存单元的地址，EA=（R），先访问寄存器，在拿着寄存器给出的去访问内存
        - 相对寻址： EA=（PC）+A，A是相对于当前指令地址的位移量，可正可负，相对于PC给出的地址
        - 基址寻址： CPU中的基址寄存器（BR）的内容加上指令格式中的形式地址A而形成的操作数的有效地址，EA=（BR）+A，基址寄存器**既可采用专用寄存器，又可采用通用寄存器。**BR不变, 适合解决动态定位的问题
        - 变址寻址： 有效地址EA等于指令字中的形式地址A与变地址寄存器IX的内容之和，即EA=（IX）+A，**可以是专用寄存器，也可是通用寄存器。**IX变，A作为基准地址不变。
        - 堆栈寻址： 由一个特定的寄存器给出，该寄存器被称为堆栈指针（SP）；寄存器堆栈又称为硬堆栈。从主存中划出一段地址，称为软堆栈

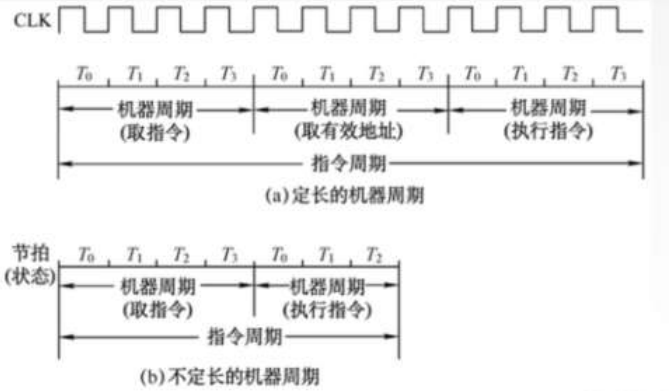
寻址方式	有效地址	访存次数(指令执行期间)
隐含寻址	程序指定	0
立即寻址	A即是操作数	0
直接寻址	EA=A	1
一次间接寻址	EA=(A)	2
寄存器寻址	EA=R_i	0
寄存器间接一次寻址	EA=(R_i)	1
转移指令 相对寻址	EA=(PC)+A	1
多道程序 基址寻址	EA=(BR)+A	1
循环程序 变址寻址	EA=(IX)+A	1
偏移寻址		

- **CISC（复杂指令系统计算机）和RISC（精简指令系统计算机）**
    - CISC：大多使用微程序控制，难以优化编译
    - RISC：大多使用硬布线控制，重视优化编译，流水线,只有Load/Store指令访存， 其余指令的操作都在寄存器之间进行

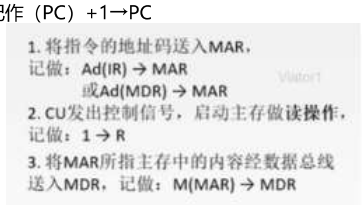
类别 / 对比项目	CISC	RISC
指令系统	复杂，庞大	简单，精简
指令数目	一般大于200条	一般小于100条
指令字长	不固定	定长
可访存指令	不加限制	只有Load/Store指令
各种指令执行时间	相差较大	绝大多数在一个周期内完成
各种指令使用频度	相差很大	都比较常用
通用寄存器数量	较少	多
目标代码	难以用优化编译生成高效的目标代码程序	采用优化的编译程序，生成代码较为高效
控制方式	绝大多数为微程序控制	绝大多数为组合逻辑控制
指令流水线	可以通过一定方式实现	必须实现

第五章<中央处理器CPU>

- **CPU的功能和基本构造**
  - CPU的功能包括
    - 指令控制：完成取指令，分析指令和执行指令的操作，即操作的顺序控制器。
    - 操作控制：CPU管理并产生由内存取出的每条指令的操作信号，把各种操作信号送往相应的部件，从而控制这些部件按指令的要求进行动作。
    - 时间控制：对各种操作加以时间上的控制。时间控制要为每条指令按时间顺序提供应有的控制信号。
    - 数据加工： 对数据进行算术和逻辑运算
    - 中断处理： 对计算机运行过程中出现的异常情况和特殊请求进行处理。
  - 控制器： 控制器是整个系统的指挥中枢。控制器有**硬布线控制器**和**微程序控制器**， 指令序列包括取指令，分析指令和执行指令
    - 控制器由程序计数器（PC），指令寄存器（IR：保存当前正在执行的那条指令），指令译码器：对操作码进行译码，存储器地址寄存器（MAR），存储器数据寄存器（MDR），时序系统和微操作信号发生器等
  - 运算器： 是接收从控制器送来的命令并执行相应的动作，对数据进行加工和处理
    - 算术逻辑单元（ALU），暂存寄存器，累加寄存器（ACC：暂存ALU的计算结果信息，实现加法），通用寄存器（如AX,BX,CX,DX,SP,存放各种操作数），程序状态字寄存器（PSW：保留运算所产生的状态信息，如：溢出标志OP，符号标志SF，零标志ZF），移位器，计数器（CT：控制乘除的操作步数）
  - 可见与不可见
    - 可见：通用寄存器组，程序状态字寄存器,PC
    - 不可见： 存储器地址寄存器，存储器数据寄存器，指令寄存器。
- **指令执行的过程**
  - 指令周期：CPU从主存中每**取出并执行一条指令所需的全部时间**
  - 指令周期常用若干机器周期来表示，一个机器周期又包含若干时钟周期（也称节拍或T周期，它是CPU操作的最基本单位
  - 取址周期 FE是为了取指令，间址周期 IND是为了取有效地址，执行周期 EX是为了取操作数，中断周期 INT是为了保存程序断点



  - 取址周期：
    现行指令地址送至存储器地址寄存器，记作PC→MAR
    向主存发送读命令，启动主存作读操作，记作1→R
    将MAR（通过地址总线）所指的主存单元中的内容（指令）经数据总线读至MDR内，记作M（MAR）→MDR
    将MDR的内容送至IR，记作MDR→IR
    指令的操作码送至CU译码，记作OP（IR）→CU
    形成下一条指令的地址，记作（PC）+1→PC



  - 间址周期： 1->R表示 读操作
  - 执行周期： 执行周期的认为是根据IR中额指令字的操作码和操作数通过ALU操作产生执行结果。不同指令的执行周期操作不同，所以没有统一的数据流向。

中断：暂停当前任务去完成其他任务。
为了能够恢复当前任务，需要保存断点。
一般使用堆栈来保存断点，这里用SP表示栈顶地址，假设SP指向栈顶元素，进栈操作是先修改指针，后存入数据。

1. CU控制将SP减1，修改后的地址送入MAR
记做：(SP)-1 → SP，(SP) → MAR
本质上是将断点存入某个存储单元，假设其地址为a，故可记做：a → MAR
2. CU发出控制信号，启动主存做写操作，
记做：1 → W
3. 将断点(PC内容)送入MDR，
记做：(PC) → MDR
4. CU控制将中断服务程序的入口地址(由向量地址形成部件产生)送入PC，
记做：向量地址 → PC

- 中断周期：
- 1> 单指令周期　2> 多指令周期　3> 流水线方案

数据通路的功能和基本结构

- 1> CPU内部单总线方式： 一条公共通路上，这种结构比较简单，但数据传输存在较多的冲突现象，性能较低。
- 2> CPU内部三总线方式： 多条公共通路上，相比之下单条总线中一个时钟内只允许传一个数据，这种方式提高了效率
- 3> 专用数据通路方式： 根据指令执行过程中的数据和地址的流动方向安排连接线路，避免使用共享总线，性能较高，但硬件量大。
- 内部总线： CPU内部连接各寄存器及运算部件之间的总线
- 系统总线： 系统总线是指同一台计算机系统的各部件，如CPU，内存，通道和各类I/O接口间相互连接的总线。
- 寄存器之间的数据传输

(PC)→Bus	PCout有效，PC内容送总线
Bus→MAR	MARin有效，总线内容送MAR

Bus指的是总线

- 主存与CPU之间的数据传送

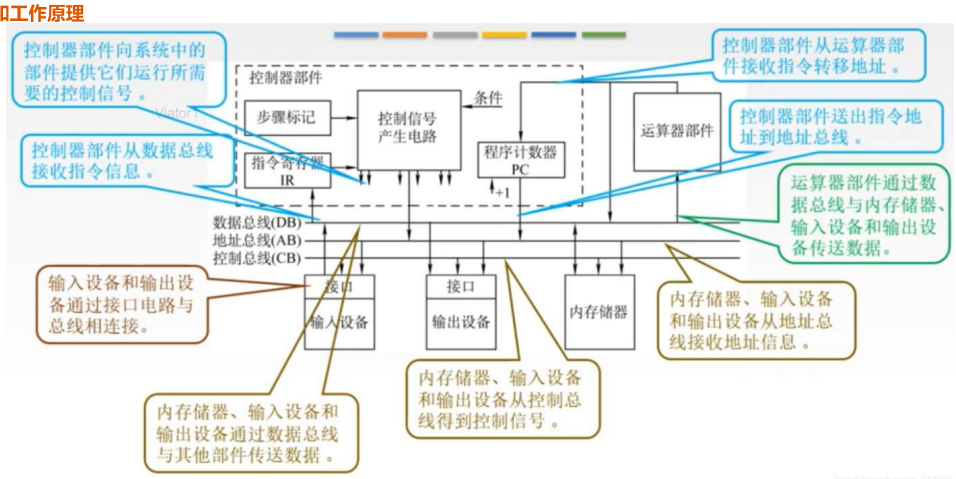
比如CPU从主存读取指令，实现传送操作的流程及控制信号为：

(PC)→Bus→MAR	PCout和MARin有效，现行指令地址→MAR
1→R	CU发读命令(通过控制总线发出，图中未画出)
MEM(MAR)→MDR	MDRin有效

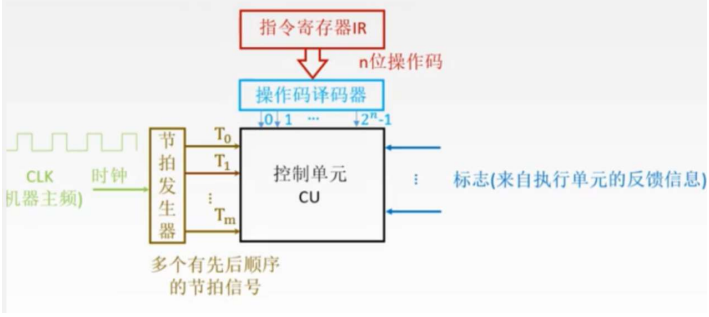
- 执行算术或逻辑运算

Ad(IR)→Bus→MAR	MDRout和MARin有效
1→R	CU发读命令
MEM(MAR)→数据线→MDR	MDRin有效
MDR→Bus→Y	MDRout和Yin有效，操作数→Y
(ACC)+(Y)→Z	ACCout和ALUin有效，CU向ALU发送加命令
Z→ACC	Zout和ACCin有效，结果→ACC

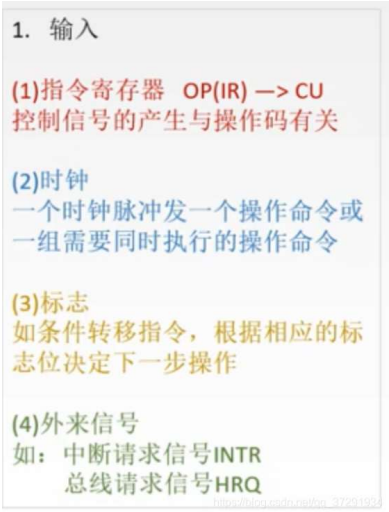
控制器的功能和工作原理



- 控制器的主要功能包括
  - 1> 从主存中取出一条指令，并指出下一条指令在主存中的位置
  - 2> 对指令进行译码或测试，产生相应的操作控制信号，以便启动规定的动作
  - 3> 指挥并控制CPU，主存，输入和输出设备之间的数据流动方向。
  - 控制器产生微操作控制信号的方式不同，控制器可分为硬布线控制器和微程序控制器
  - 硬布线控制器：基本原理是 根据指令的要求，当前的时序及外部和内部的状态，按时间的顺序发送一系列微操作控制信号，它由复杂的组合逻辑门电路和一些触发器构成，因此又称为组合逻辑控制器
    - 指令的操作码是决定控制单元发出不同操作命令（控制信号）的关键。为了简化控制单元（CU）的逻辑，将指令的操作码译码和节拍发生器从CU分离出来，便可得到减缓的控制单元图

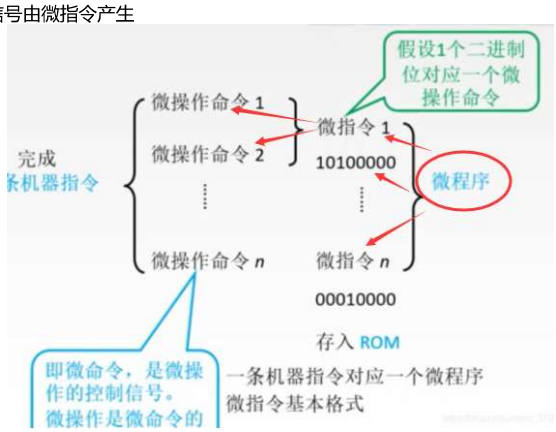


- CU的输入信号来源



- CU的输出信号

  - 硬布线控制器的时序系统及微操作
    - 1> 时钟周期。用时钟信号控制节拍发生器，可以产生节拍，每个节拍的宽度正好对应一个时钟周期。
    - 2> 机器周期。 常把一条指令的执行过程划分为若干个阶段（如取指、译码、执行等），每一阶段完成一个基本操作。完成一个基本操作所需要的时间称为机器周期
    - 3> 指令周期
    - 4> 微操作命令分析。控制单元具有发出各种操作命令（控制信号）序列的功能。这些命令与指令有关，而且必须按一定次序发出，才能使机器有序地工作。
  - CPU的控制方式
    - 1> 同步控制方式。系统有一个统一的时钟，所有的控制信号均来自这个统一的时钟信号。通常以最长的微操作作序列和最烦琐的微操作作为标准。
    - 2> 异步控制方式。异步控制方式不存在基准时标信号，各部件按自身固有的速度工作，通过应答方式进行联络。
    - 3> 联合控制。是介于同步，异步之间的一种折中。大部分采用同步控制，小部分采用异步控制的办法。
- 微程序控制器
  - 微程序控制器采用存储逻辑实现，把微操作信号代码化，使每条机器指令转换成为一段微程序并存入一个专门的存储器（控制存储器）中，微操作的控制信号由微指令产生



  - 微指令包含2部分信息
    - 1> 操作控制字段，又称微操作码字段，用于产生某一步操作所需的各种操作控制信号。
    - 2> 顺序控制字段，又称为地址码字段，用于控制产生下一条要执行的微指令地址。

    - 若指令系统中具有n种机器指令，则控制存储器中的微程序数至少是n+1（1为公共的取值微程序）
    - 微指令的格式
      - 1> 水平型微指令： 指令字中的一位对应一个控制信号

      - 2> 垂直型微指令

2. 垂直型微指令　类似机器指令操作码的方式，由微操作码字段规定微指令的功能。

基本格式

优点：微指令短、简单、规整，便于编写微程序；
缺点：微程序长，执行速度慢，工作效率低。

- 3> 混合型微指令

3. 混合型微指令　在垂直型的基础上增加一些不太复杂的并行操作。

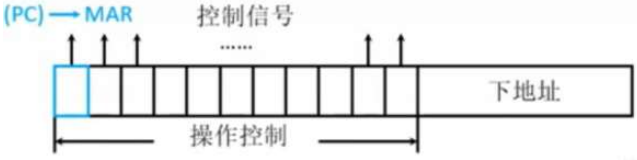
微指令较短，仍便于编写；微程序也不长，执行速度加快。

- 微指令的译码方式
  - 1> 直接编码（直接控制）方式：无须进行译码，微指令的微命令字段中每一位都代表一个微命令

(1) 直接编码（直接控制）方式

在微指令的操作控制字段中，每一位代表一个微操作命令

某位为“1”表示该控制信号有效



  - 2> 字段直接编码方式：分成若干的段

(2) 字段直接编码方式

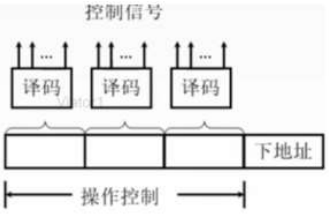
将微指令的控制字段分成若干“段”，每段经译码后发出控制信号

微命令字段分段的原则：

① 互斥性微命令分在同一段内，相容性微命令分在不同段内。

② 每个小段中包含的信息位不能太多，否则将增加译码线路的复杂性和译码时间。

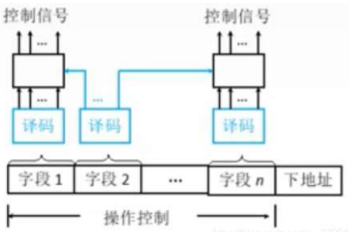
③ 一般每个小段还要留出一个状态，表示本字段不发出任何微命令。因此，当某字段的长度为3位时，最多只能表示7个互斥的微命令，通常用000表示不操作。



  - 3> 字段间接编码方式

(3) 字段间接编码方式

一个字段的某些微命令需由另一个字段中的某些微命令来解释，由于不是靠字段直接译码发出的微命令，故称为字段间接编码，又称隐式编码。



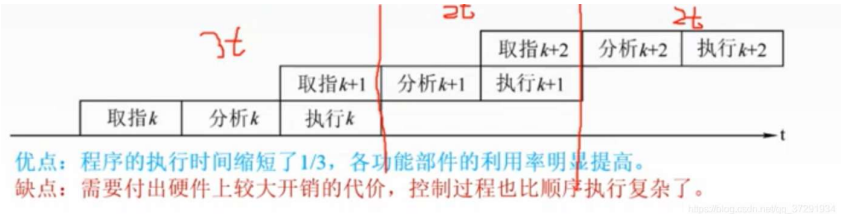
- 微指令的地址形成方式
  - 1> 直接由微指令的下地址字段指出。微指令格式中设置一个下地址字段，由微指令的下地址字段直接指出后继微指令的地址，又称断定方式
  - 2> 根据机器指令的操作码形成。机器指令取至指令寄存器后，微指令的地址由操作码经微地址形成部件形成

类别 / 对比项目	微程序控制器	硬布线控制器
工作原理	微操作控制信号以微程序的形式存放在控制存储器中，执行指令时读出即可	微操作控制信号由组合逻辑电路根据当前的指令码、状态和时序，即时产生
执行速度	慢	快
规整性	较规整	烦琐、不规整
应用场合	CISC CPU	RISC CPU
易扩充性	易扩充修改	困难

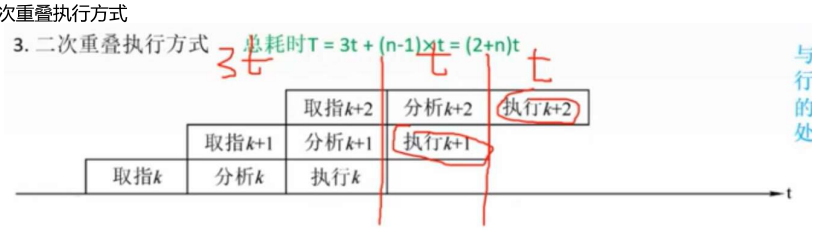
- **指令流水线**
  - 基本概念： 把一个重复的过程分解为若干子过程，每个子过程与其他子过程并行执行。 流水线技术只需要增加少量硬件就能把计算机的运算速度提高几倍，所以是计算机中普遍使用的一种并行技术。
  - 一条指令的执行过程可以分为多个阶段，如果采用三个阶段，就有取指，分析，执行。
  - 当多条指令在处理器中执行时，可以采用三种方式：
    - 1> 顺序执行方式

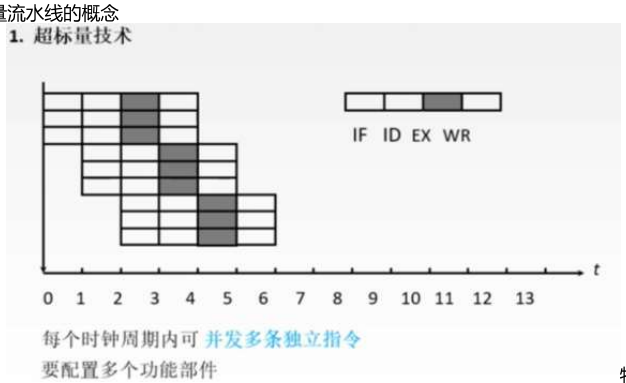
传统冯·诺依曼机采用顺序执行方式，又称串行执行方式。
优点：控制简单，硬件代价小。
缺点：执行指令的速度较慢，在任何时刻，处理机中只有一条指令在执行，各功能部件的利用率很低。

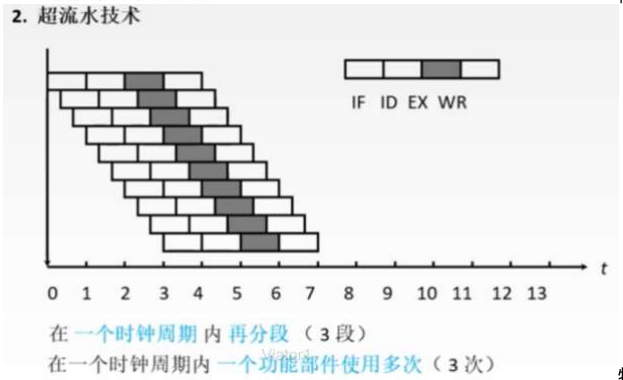
    - 2> 一次重叠执行方式



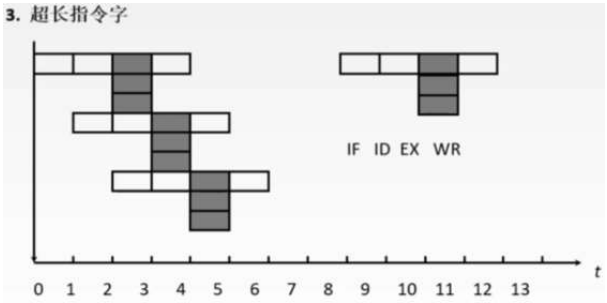
- 3> 二次重叠执行方式



- 流水线分类
  - 部件功能级，处理机级和处理机间级流水线
  - 单功能流水线和多功能流水线（通过各段间不同的连接方式实现）
  - 静态流水线和动态流水线
    - 静态：同一时间内，各段只能按同一功能的连接方式工作。
    - 动态：同一时间内，某些段正在实现某种运算，另一些段却正在进行另一种运算。效率高，但是控制复杂。
  - 线性流水线和非线性流水线（看功能间是否有反馈信号）
    - 线性：从输入到输出，每个功能只允许经过一次，不允许反馈回路。
    - 非线性：存在反馈回路，从输入到输出，某些功能段将数次通过流水线，这种流水线适合进行线性递归的运算。
- 影响流水线的因素
  - 1> 结构相关（资源冲突）
  - 由于多条指令在同一时刻争用同一资源而形成的冲突称为结构相关
  - 2> 数据相关（数据冲突）
  - 数据相关是在一个程序中，存在必须等前一条指令执行完才能执行后一条指令的情况
  - 3> 控制相关（控制冲突）
  - 当流水线遇到转移指令和其它改变PC值得指令而造成断流时，会引起控制相关。
- 流水线的性能指标
  - 1> 流水线的吞吐率=单位时间内完成的任务的数量/所用时间T
  - 2> 流水线的加速比=不用流水线所以的时间/用流水线所用的时间
  - 3> 流水线的效率=时空区中有效面积/时空区的总面积
- 超标量流水线的概念
  - 

特点：一个时钟周期内可并发堕多条独立指令(同时发)
  - 

特点：一个时钟周期内可分段发多条独立指令（分段发）



第六章<总线>

- **总线的概念**：解决I/O设备和主机之间连接的灵活性问题，计算机的结构从分散连接发展为总线连接； 分时和共享
  - 总线上的 主设备和从设备
    - 主设备： 指获得总线控制权的设备。
    - 从设备：指被主设备访问的设备，它只能响应从主设备发送的各种总线命令。
  - 总线的特性
    - 机械特性（尺寸，形状）
    - 电气特性（传输方向和有效的电平范围）
    - 功能特性（每根传输线的功能）
    - 时间特性（信号和时序的关系）
      - 在一个总线周期内传输存储地址连续的多个数据字的总线传输方式，称为猝发传送，及一次传输一个地址和一批地址连续的数据。
  - 总线的分类
    - 1> 片选总线：（CPU内部总线）它是CPU芯片内部寄存器与寄存器之间，寄存器与ALU之间的公共连接线。
    - 2> 系统总线：系统总线是计算机系统内各功能部件（CPU，主存，I/O接口）之间相互连接的总线；按系统总线传输信息内容的不同，又可以分为：数据总线，地址总线，控制总线
    - 3> 通信总线：通信总线是计算机系统之间或计算机系统与其他系统（如远程通信设备，测试设备）之间传送信息的总线。
      - 串行：成本低，远距离。
      - 并行：逻辑时序简单，电路实现容易；信号线数量多，成本高，占用布线空间。
  - 系统总线的结构(不是cpu内部，注意区分)
    - 1> 单总线结构：单总线结构将CPU，主存，I/O设备都挂在一组总线上（ 单总线并不是只有一根信号线，系统总线传送信息的不同可细分为地址总线，数据总线和控制总线）
    - 2> 双总线结构：有两条总线，一条是主存总线，用于在CPU，主存和通道之间传送数据；另一条是I/O总线，用于在多个外部设备与通道之间传送数据。
    - 3> 三总线结构：分别有主存总线，I/O总线和直接内存访问（DMA）总线
  - 总线的性能指标
    - 传输周期（总线周期）：包括申请，寻址，传输，结束阶段，通常由若干个总线时钟周期构成
    - 时钟周期：机器的时钟周期，总线也受此时钟的控制
    - 工作频率：总线周期的倒数 若总线周期=B个时钟周期，则总线的工作效率=时钟频率/N
    - 时钟频率：时钟周期的倒数
    - 总线宽度：又叫 总线位宽，是指总线上能同时传输的数据位数
    - 总线带宽：总线的传输速率，即单位时间内，总线上能传输数据的位数。
      - 总线带宽 = 总线的工作频率 * 总线宽度（bit/s）= 总线工作频率 * （总线宽度/8）（B/s）
- **总线的仲裁**： 解决多个主设备同时竞争总线控制权的问题, 以某种方式选择一个主设备优先获得总线控制权。
  - 集中仲裁方式和分布仲裁方式
    - 集中仲裁方式： 总线控制逻辑基本上集中于一个设备（如CPU）中。将所有总线请求集中起来，利用一个特定的裁决算法进行裁决，称为集中仲裁方式。集中仲裁方式有链式查询方式，计数器定时查询方式和独立请求方式。（三类线：**请求**，**允许**，**总线忙**）

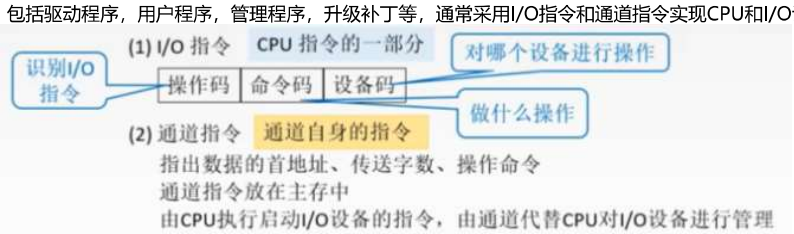
仲裁方式 / 对比项目	链式查询	计数器定时查询	独立请求
控制线数	3 总线请求：1 总线允许：1 总线忙：1	$\lceil \log_2 n \rceil + 2$ 总线请求：1 总线允许：$\lceil \log_2 n \rceil$ 总线忙：1	2n+1 总线请求：n 总线允许：n 总线忙：1
优点	优先级固定 结构简单，扩充容易	优先级较灵活	响应速度快 优先级灵活
缺点	对电路故障敏感 优先级不灵活	控制线较多 控制相对复杂	控制线多 控制复杂

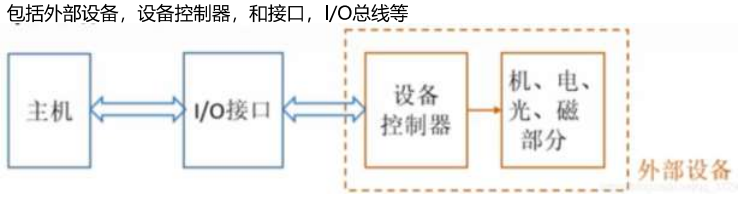
“总线忙”信号的建立者是获得总线控制权的设备

    - 分布仲裁方式：分布仲裁方式不需要中央仲裁器，每个潜在的主模块都有自己的仲裁号和仲裁器。当它们有总线请求时，就会把它们各自唯一的仲裁号发送到共享的仲裁总线上，每个仲裁器将从仲裁总线上得到的仲裁号与自己的仲裁号进行比较。若仲裁总线上的仲裁号优先级高，则它的总线请求不予响应，并撤销它的仲裁号。最后，获胜者的仲裁号保留在仲裁总线上。
- **总线操作和定时**
  - 总线定时： 总线定时是指总线在双方交换数据的过程中需要时间上的配合关系的控制，这种控制称为总线定时，其实质是一种协议或规则，主要有同步和异步两种基本的定时方式
    - 同步定时方式： 用一个统一的时钟信号来协调发送和接收双方的传送定位关系， 时钟产生相等的时间间隔，每个间隔够构成一个总线周期。 在一个总线周期中，发送方和接收方可以进行一次数据传送。
    - 异步定时方式： 没有统一的时钟，也没有固定的时间间隔，完全依靠传送双方相互制约的“握手”信号来实现定时控制。速度比同步定时要慢
  - 总线传输的阶段
    - 1> 申请分配阶段。由需要使用总线的主模块（或主设备）提出申请，经总线仲裁机构决定将下一传送周期的总线使用权授予某一申请者。也可将此阶段细分为传输请求和总线仲裁两个阶段。

        - 2> 寻址阶段。取得使用权的主模块通过总线发出本次要访问的从模块（或从设备）的地址及有关命令，启动参与本次传输的从模块。
        - 3> 传输阶段。主模块和从模块进行数据交换，可单向或双向进行数据传送。
        - 4> 结束阶段。主模块的有关信息均从系统总线上撤除，让出总线使用权。
    - 请求和回答的信号撤销是否互锁？分类
        - 1> 不互锁方式。主设备发出“请求”信号后，不必等到接到从设备的“回答信号”，而是经过一段时间便撤销“请求”信号。发“回答”信号同理。
        - 2> 半互锁方式。主设备在发出“请求”信号后，必须在接到从设备“回答”后，才撤销“请求”信号，有互锁关系。但从设备发出“回答”信号和上面不互锁方式相同。
        - 3> 全互锁方式，主设备在发出“请求”信号后，必须在接到从设备“回答”后，才撤销“请求”信号；从设备发出“回答”信号后，必须在获知主设备“请求”信号已撤销后，在撤销其“回答”信号。双方存在互锁关系。
- **总线标准**
    - 1> ISA（industry Standard Architecture，工业标准体系结构）系统总线
    - 2> EISA 系统总线
    - 3> PCI 局部总线 32bit 即插即用 突发 多路复用
    - 4> AGP 局部 局部 3D显卡
    - 5> PCI-Express 点对点**串行** 全双工 热插拔
    - 6> VESA 局部 高速传输活动图像大量数据
    - 7> USB 设备总线，串行
    - 8> RS-232C 串行 通信总线 是数据终端设备和数据通信设备
    - 9> IDE（ATA） 硬盘光驱接口
    - 10> SATA 串行硬盘接口
    - 11> SCSI 智能通用接口
    - 12> PCMCIA 便携设备接口 能耗低个人电脑
-

第七章<输入\输出设备>

- **I/O系统基本概念**
    - I/O软件：包括驱动程序，用户程序，管理程序，升级补丁等，通常采用I/O指令和通道指令实现CPU和I/O设备的信息交换
        - 

    - I/O硬件：包括外部设备，设备控制器，和接口，I/O总线等
        - 

    - I/O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。
        - 2> **程序中断方式**。只在I/O设备准备就绪并向CPU发出中断请求时才给予响应
        - 3> **DMA方式**。主存和I/O设备之间有一条直接数据通路，当主存和I/O设备交换信息时，无须调用中断服务程序。
        - 4> **通道方式**。在系统中设有通道控制部件，每个通道都挂接若干外设，主机在执行I/O命令时，只需启动有关通道，通道将执行通道程序，从而完成I/O操作。
- **外部设备**
    - 输出屏幕：CRT（阴极射线管显示器），LED，LCD
        - 容量 = 分辨率 * 灰度级位数
        - 带宽 = 分辨率 * 灰度级位数 * 帧频
    - 打印机
        - 击打式和非击打式
        - 点阵式打印机，针式打印机，喷墨式打印机，激光打印机
    - 外存储器
        - 磁盘存储器
            - 扇区（块）是磁盘读写的最小单位
            - 磁头数：即记录面数
            - 柱面数：表示一面盘面上有多少条磁道
            - 面密度 = 道密度 * 位密度
            - 平均存取时间 = **寻道时间**（磁头移动）+ **旋转延迟时间**（磁头定位到所在扇区：2分之一圈）+ **传输时间**（传输数据所花费的时间）
            - 数据传输率 = 单位时间内向主机传送数据的字节数
    - 磁盘地址
        - 

    - 磁盘阵列

RAID的分级如下所示。在RAID1～RAID5的几种方案中，无论何时有磁盘损坏，都可以随时拔出受损的磁盘再插入好的磁盘，而数据不会损坏。

RAID0：无冗余和无校验的磁盘阵列。

RAID1：镜像磁盘阵列。

RAID2：采用纠错的海明码的磁盘阵列。

RAID3：位交叉奇偶校验的磁盘阵列。

RAID4：块交叉奇偶校验的磁盘阵列。

RAID5：无独立校验的奇偶校验磁盘阵列。

- 光盘存储器

光盘的类型如下：

CD-ROM：只读型光盘，只能读出其中内容，不能写入或修改。

CD-R：只可写入一次信息，之后不可修改。

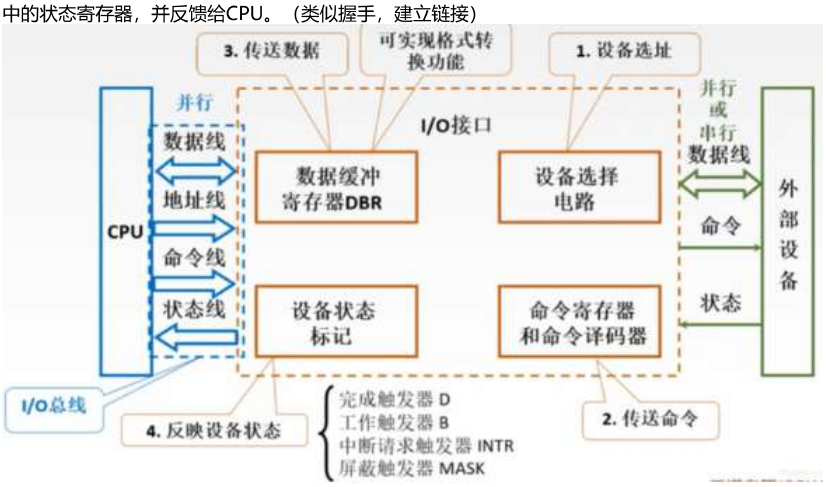
CD-RW：可读可写光盘，可以重复读写。

DVD-ROM：高容量的CD-ROM，DVD表示通用数字化多功能光盘。

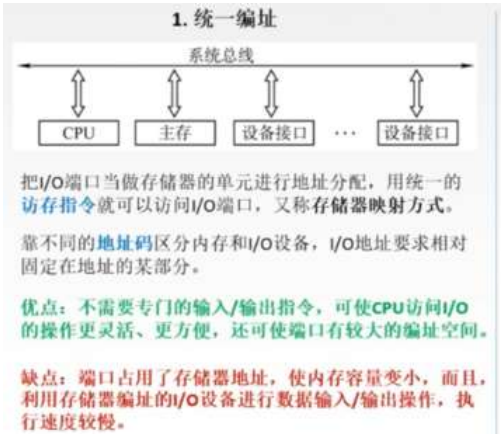
- 固态硬盘
  - 微小型高档笔记本计算机采用高性能Flash Memory 作为硬盘来记录数据，这种“硬盘”称为固态硬盘。固态硬盘除需要Flash Memory外，还需要其他硬件和软件的支持。

I/O接口

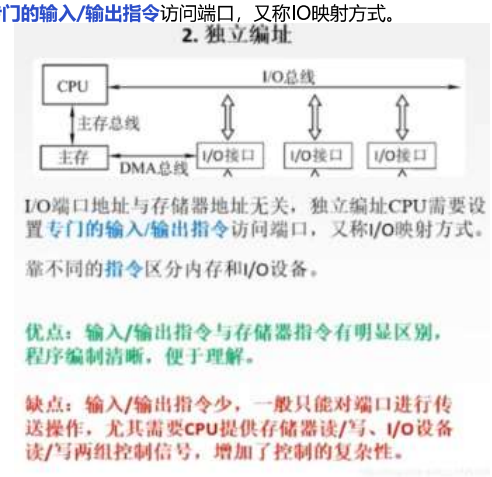
- I/O接口（I/O控制器）是主机和外设之间的交接界面，主机和外设具有各自的工作特点，它们在信息形式和工作速度上具有很大的差异，接口正是为了解决这些差异设置的。
- I/O接口的功能
  - 实现主机和外设的通信联络控制。
  - 进行**地址译码**和**设备选择**。CPU送来选择外设的地址码后，接口必须对地址进行译码以产生设备选择信息，使主机能和指定外设交换信息。
  - **数据缓冲**。CPU与外设之间的速度往往不匹配，为了消除速度差异，接口必须设置数据缓冲寄存器，用于数据的暂存，以避免因速度不一致而丢失数据
  - **信号格式的转换**。外设与主机两者的电平，数据格式都可能存在差异，接口应提供计算机与外设信号格式的转换功能，如电平转换，并/串或串/并转换，模/数或数/模转换等
  - **传送控制命令和状态信息**。CPU要启动某一外设时，通过接口中的命令寄存器向外设发出启动命令；外设准备就绪时，将“准备好”状态信息送回接口中的状态寄存器，并反馈给CPU。（类似握手，建立链接）



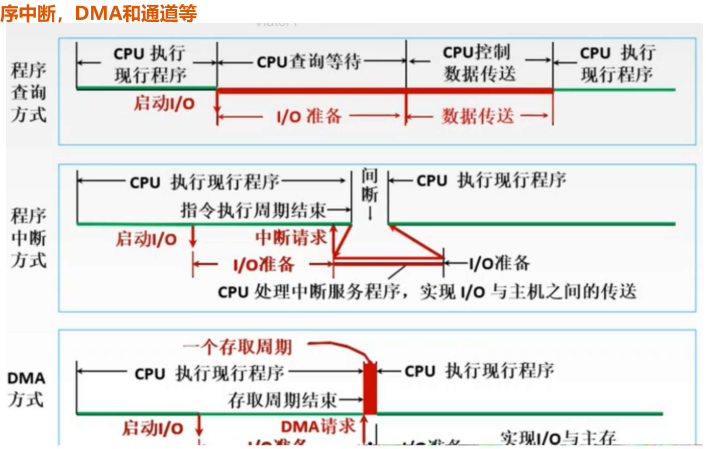
- I/O接口的基本结构
  - 1> **内部接口**：内部接口与系统总线相连，实质上是与内存，CPU相连。数据的传输方式只能是**并行传输**。
  - 2> **外部接口**：外部接口通过接口电缆与外设相连，外部接口的数据传输可能是串行方式，因此I/O接口需具有**串/并**转换功能。
  - 注意：接口和端口是两个不同的概念。端口是指接口电路中可以进行读/写的寄存器，**若干端口加上相应的控制逻辑才可以组成接口**。
- I/O接口的类型
  - 1> 按数据传送方式可分为并行接口和串行接口。
  - 2> 按主机访问I/O设备的控制方式可分为**程序查询接口，中断接口**和**DMA接口**。
  - 3> 按功能选择的灵活性可分为可编程接口和不可编程接口。
- I/O**端口**及其编址（多个端口+对应的控制逻辑才可以组成接口）
  - I/O端口：接口电路中可被CPU直接访问的**寄存器，**主要有**数据**端口，**状态**端口和**控制**端口，若干端口加上相应的**控制逻辑电路**组成接口。
  - I/O端口要想能够被CPU访问，必须要有端口地址，每个端口对应一个端口地址。而I/O端口的编址方式有与存储器统一编址和独立编址两种：
    - 存储器统一编址：用统一的**访存指令**，又称作 存储器映射方式。靠不同的地址码区分内存和IO设备



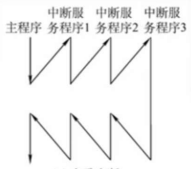
- 独立编址：CPU设置**专门的输入/输出指令**访问端口，又称IO映射方式。



- **I/O方式：常用的有程序查询，程序中断，DMA和通道等**

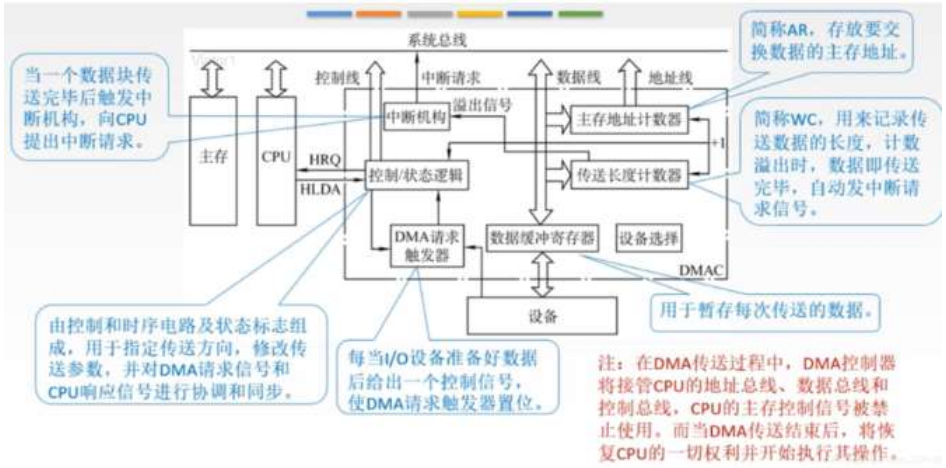


- **程序查询方式**：信息交换的控制完全由主机执行程序实现，程序查询方式的接口中设置一个数据缓冲寄存器（数据端口）和一个设备转态寄存器（状态端口）。
- **程序中断方式**
  - 1> 中断请求
    - 中断请求是指中断源向CPU发送中断请求信号。每个中断源向CPU发送中断请求的时间是随机的。为记录中断事件并区分不同的中断源，中断系统需对每个中断源设置**中断请求标记触发器INTR**，当其状态为“1”时，表示中断源有请求
  - 2> 中断判优
    - 中断判优既可以用硬件实现，有可用软件实现。硬件实现是通过硬件排队器实现的。软件实现是通过查询程序实现的。
  - 3> CPU响应中断的条件
    - 1.中断源有中断请求　2.CPU允许中断及开中断　3.一条指令执行完毕，且没有紧迫的任务。
  - 4> 中断隐指令
    - CPU响应中断后，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。我们将它成为中断隐指令。
  - 5> 中断向量
    - 不同设备有不同的中断服务程序，每个中断服务程序都有一个入口地址，CPU必须找到这个入口地址，即中断向量。



	单重中断	多重中断
中断隐指令	关中断	关中断
	保存断点（PC）	保存断点（PC）
	送中断向量	送中断向量
中断服务程序	保护现场	保护现场和屏蔽字
	-	开中断
	执行中断服务程序	执行中断服务程序
	-	关中断
	恢复现场	恢复现场和屏蔽字
	开中断	开中断
	中断返回	中断返回

- **DMA方式**
  - DMA方式是一种**完全由硬件**进行成组信息传递的控制方式，它具有程序中断方式的优点。DMA方式在外设和内存之间开辟一条“**直接数据通路**”信息传送不再经过CPU，降低了CPU在传送数据时的开销，因此称为**直接存储器存取方式**。DMA传送的过程中，DMA控制器接管CPU的地址总线，数据总线和控制总线。



  - 解决DMA和CPU访存冲突
    - 周期挪用：在CPU不访问存储器的那些周期实现DMA操作。