基本组成

2022年6月16日 17:11

计算机系统分为

硬件：计算机的实体，如主机、外设等

软件：具有各类特殊功能的程序

系统软件：用来管理计算机系统

操作系统

语言处理程序

服务性程序

数据库管理系统

应用软件

按任务需要而编写的程序

编译型语言使用编译程序（如C++），解释型语言使用解释程序（如Python）

微指令：将一条机器语言进一步分割，把不同时间段执行的步骤放到不同微指令当中

高级语言 > 汇编语言 > 操作系统 > 机器语言 > 微指令系统

步骤为：

1、由解释器或者编译器将高级语言转化为汇编语言

2、用汇编语言翻译成机器语言

3、用机器语言解释操作系统　　1、2、3为软件实现

4、用微指令解释机器语言

5、硬件直接执行微指令　　4、5为硬件实现

即计算机系统构成可表示为

用户>应用程序>操作系统>裸机

冯·诺伊曼计算机的6个特点：

有五大组成部分

1、控制器（CU，指挥程序运行）

2、计算器（ALU，包括算术运算和逻辑运算）　1、2结合即为CPU

3、存储设备

主存（内存）

辅存（硬盘）　注意辅存算 I/O设备

4、输入

5、输出　4、5组成 I/O设备

指令和数据以同等的地位存储于存储器，可以按地址访问

指令和数据用二进制表示

指令由操作码（怎么操作）和地址码（操作数在哪里）组成

以运算器为中心

存储程序

存储器的基本组成　　注意！这里存储器单指内存

存储体-存储单元-存储元件

MAR：存储器地址寄存器，反映存储单位的个数，其具体数值为2的指数

MDR：存储器数据寄存器，反映存储字长（位数）

位数就是一个数由多少个数字，1234就是4位

如MAR=10,MDR=8，2^10 =1k 表示能存储1k的8位数据，k在这应该是1千个数据的意思

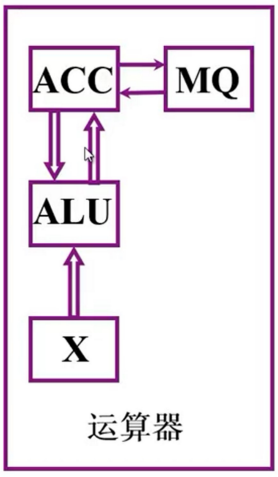
运算器的基本组成

ALU逻辑运算单元

X运算的输入

ACC运算的输入与结果的保存

MQ另一个寄存器（只在乘除法中用到）

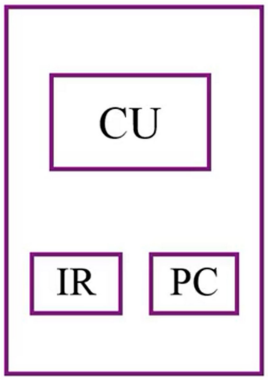


	ACC	MQ	X
加法	被加数 和		加数
减法	被减数 差		减数
乘法	乘积高位	乘数 乘积低位	被乘数
除法	被除数 余数	商	除数

控制器的功能

解释指令（取指令和分析指令）

保证指令按序执行



完成一条指令：取指令 PC；分析指令 IR；执行指令 CU

PC 存放当前欲执行指令的地址，具有计数功能（PC）+1→PC

IR 存放当前欲执行的指令

计算机主要技术指标

机器字长：cpu一次能处理数据的位数，与cpu中寄存器位数相等

如果寄存器是8位，要完成64位加法运算就要运算8次，而64位的只需一次加法

运算速度

主频（不直接）

核心数与核心的线程数

吉普森法：计算指令出现频率的加权平均值的运算速度

CPI：执行一条指令需要的时钟周期（仍然需要加权平均）

IPC：和上面的类似，指每秒能执行多少条指令

FLOPS：每秒浮点运算次数

存储容量

内存

存储单元个数*存储字长=总存储量

字节数（多少KB,GB等）

辅存

字节数

总线

2022年6月16日 23:16

总线（bus）：连接计算机各个部件的信息传输线，是各个部件共享的传输介质
特点：可以连接许多设备，但同一时刻只能有一对设备进行数据交换
总线上的信息传送
串行：一个个传送，就像只能通过一辆车的公路，汽车只能一个个跑
串行总线本身频率就比并行总线更高，传输效率更高
所以高速远距离传输串行是首选
并行：多位传送，好比开了几条道的公路，多辆汽车一起跑
但是并行需要多条线路，且传输数据较远的话，多条平行线路之间会出现数据干扰
且并行口各线路需要同时发送同时接收，每个延迟都会影响整体速度
并行想要传输远一点，就需要特殊的技术

面向CPU的双总线结构
I/O总线：连接IO接口以及各种IO设备
M总线：连接主存（内存）
面向存储器的双总线结构
系统总线：主存和I/O设备
存储总线：cpu和主存的专用总线

注意：双总线结构中两条总线仍然不能同时使用
但是两条总线速度不同，可以灵活应对不同场景

总线的分类
按位置分类
片内总线：芯片内部的总线
系统总线：计算机各部件之间的信息传输
数据总线：双向的，传输数据
地址总线：单向的，寻址
控制总线：传输控制信号，有入有出
通信总线：计算机和其他计算机设备进行信息传输，有串行和并行两种传输方法

总线的特性
机械特性：尺寸、形状、管脚数及排列顺序
电气特性：传输方向和有效电平范围
功能特性：每根传输线的功能
时间特性：信号之间的时序关系

总线的性能指标
总线宽度：数据线的根数
标准传输率：每秒传输的最大字节数（Mbps百万字节）

时钟同步/异步：同步、不同步
总线复用：地址线和数据线复用
信号线数：地址线、数据线、控制线的总和
总线控制方法：突发、仲裁、计数等
其他指标：负载能力等

总线的结构
单总线结构
双总线结构
三总线结构
IO总线
DMA总线
主存总线
或者
局部总线
系统总线
扩展总线
四总线结构

总线控制
总线判优控制：解决由哪个主设备来使用总线
1、主设备：对总线有控制权
2、从设备：响应从主设备发来的总线命令
总线判优控制
集中式
链式查询
计数器定时查询
独立请求方式
分布式
总线通信控制
目的：解决通信双方协调配合的问题
总线传输周期
申请分配阶段
寻址阶段
传输数据阶段
结束阶段
总线通信四种方式
同步通信：由统一时标控制数据传送
异步通信：采用应答式，而没有统一时标
半同步通信：同步与异步结合

分离式通信：发挥总线最大效率

存储器

2022年6月17日　9:30

存储器的分类

- 按存储介质
  - 半导体存储器：如闪存。数据易失，SSD用的Flash Memory技术
  - 磁表面存储器（磁盘）
  - 磁芯存储器
  - 光盘存储器
- 按存取方式
  - 存取时间和物理地址无关（随机访问）
    - 随机存储器：在程序执行过程中可读可写
    - 只读存储器：程序执行时只读
  - 存取时间与物理地址有关（串行访问）
    - 顺序存取存储器（磁带）
    - 随机存取存储器（磁盘）
- 按在计算机中的作用
  - 主存储器
    - 随机存取存储器RAM：可读可写
    - 只读存储器ROM：只读
  - 高速缓冲存储器
  - 辅存储器
    - 硬盘

寄存器：

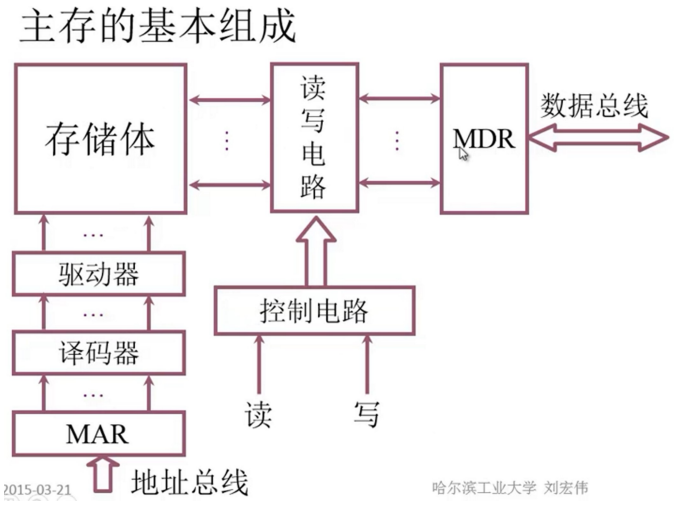
CPU内部和其他部件 用来存放数据的一些小型存储区域，用来暂时存放参与运算的数据和运算结果

固态硬盘用到的颗粒也是基于NAND FLASH技术（由ROM发展而来），和u盘以及手机存储有点相似，所以说硬盘和ROM还是有关系的，固态硬盘的存储颗粒是ROM技术发展的产物，但不能说ROM就是硬盘

存储器的层次结构

- 缓存-主存、主存-辅存

主存的基本结构



主存的技术指标

- 存储容量
- 存储速度

高速缓冲存储器Cache

使用缓存的原因：CPU和内存之间速度剪刀差越来越大

主存容量大速度低，Cache容量小速度高

程序访问的局部性原理

时间的局部性：当前在使用的指令和数据将来还将被使用

空间的局部性：当前使用的指令和数据的相邻的数据与指令也可能被使用

Cache的工作原理

主存和缓存按块存储

块的大小（块长）相同，所以主存的块数量要远大于缓存的块的数量

缓存的命中与未命中

命中：cpu需要主存的数据时，该数据已经调用在缓存中了

称为主存块与缓存块建立了对应关系

未命中：主存块未调用在缓存中，cpu必须从主存中找数据

因此命中率非常重要

cache的命中率：cpu欲访问的信息在缓存中的比重

命中率与缓存的容量和块长有关

缓存-主存系统的效率

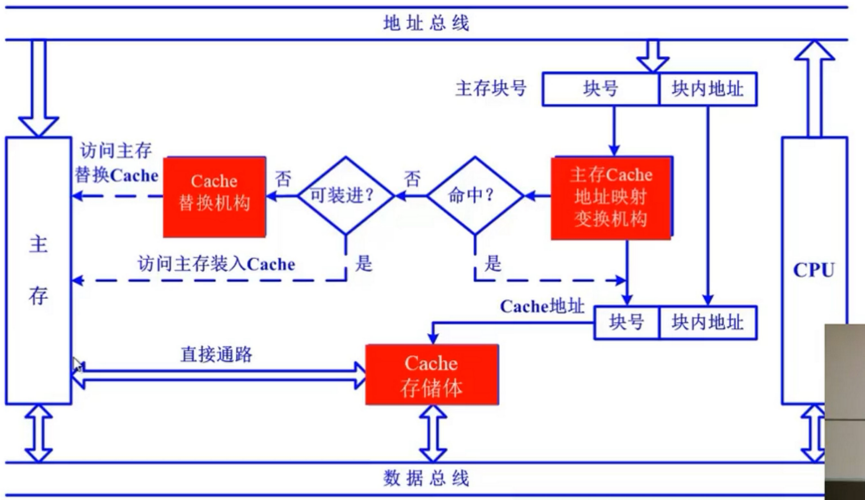
效率e与命中率有关

e=访问cache的时间/平均访问时间*100%

设 Cache 命中率 为 h，访问 Cache 的时间为 t_c，
访问 主存 的时间为 t_m

则 $$e = \frac{t_c}{h \times t_c + (1-h) \times t_m} \times 100\%$$

缓存的结构



Cache的读和写操作

Cache的改进

1、增加缓存的级数

片内cache（cpu内部）

片外cache

2、统一缓存和分立缓存

指令cache + 数据cache，避开了指令和地址同时存放的缺陷

辅助存储器

特点：不直接与cpu交换信息

输入输出系统

2022年6月17日 19:13

发展概况

1、早期：分散连接

2、接口模块和DMA阶段

总线连接，cpu和IO设备并行工作

3、具有通道结构的阶段

4、具有IO处理机的阶段

使用IO处理机来控制IO设备

输入输出系统的组成

1、IO软件

IO指令：CPU指令的一部分

操作码-命令码-设备码

通道指令：通道自身的指令

通道有自己的控制器，有的还有自己的存储器

指出数组的首地址、传输字操作命令

2、IO硬件

方式1：设备-IO接口

方式2：设备-设备控制器-通道

IO设备和主机的联系方式

CPU要想联系到IO设备，就得先知道IO设备在哪（地址）

首先：IO设备编址

然后：设备选址

最后：使用串行或并行进行传送

IO设备和主机的联络方式

1、立即响应

2、异步工作（应答模式）

并行

串行

IO设备和主机的连接方式

1、辐射式连接：每台外设都有一套线和主机相连不便于增删设备

2、总线连接

IO设备和主机信息传送的控制方式

1、程序查询方式

2、程序控制方式

3、DMA方式

4、通道方式

5、IO处理机方式

IO接口

组成

功能

类型

DMA（Direct Memory Access）控制器是一种在系统内部转移数据的独特外设，可以将其视为一种能够通过一组专用总线将内部和外部存储器与每个具有DMA能力的外设连接起来的控制器。它之所以属于外设，是因为它是在处理器的编程控制下来 执行传输的

计算机的运算方法

2022年6月17日 19:56

无符号数

没有正负号等符号的数值

保存和表示非常简单

寄存器位数表示无符号数的范围：如8位寄存器，可以表示00000000到11111111也就是0到255的数

有符号数

包含数值部分和符号部分

机器数：保存在计算机里的数，符号用数字来表示的数

真值：我们日常使用的带符号的数，如-8

1表示负数，0表示正数

表示数据的方法：

原码表示法、反码表示法、补码表示法

在一般存储数据的时候，用8个二进制数来存储数据，形式为

xxxx xxxx，称之为原码

第一个x为符号位，等于0时为正数，等于1时为负数。比如1000 1000表示-8，0000 0011表示3，1000 0011表示-3。很明显，-3+3=0，但是在原码上直接相加却不是这个结果

由于原码正负数直接相加不能得出正确结果，为此引入反码的概念。正数的反码就是原码，负数的反码=原码除符号位以外的按位取反，即0改成1，1改成0。而正负数就是反码相加（仍需要逢2进1），然后将反码取回原码即可，注意正负数的差别

如-1 + 1 =0

即1000 0001 + 0000 0001 = 1111 1110 + 0000 0001 = 1111 1111，再取回原码即1000 0000

但是一个正数加其对应的负数的时候，符号位为1，即结果为-0而不是0

这时引入补码的概念。补码在计算机内应用非常广泛

正数的补码即原码，而负数的则是反码+1

即1111 1111 + 0000 0001 = （1） 0000 0000，这里多出来的（1）自然去掉

此外还有移码表示法

而对于小数，有两种约定俗成的给出方法，一种是将小数点放在符号位后边，另一种是放在数值位的最后边

数的定点表示和浮点表示

定点表示

1、小数点放在符号位后边——小数定点计算机

2、小数点放在数值位后边——整数定点计算机

浮点表示

引入浮点数表示的原因

1、编程困难，程序员需要调整小数点的位置

2、数的表示范围小，为了能表示两个大小相差很大的数据，需要很长的机器字长

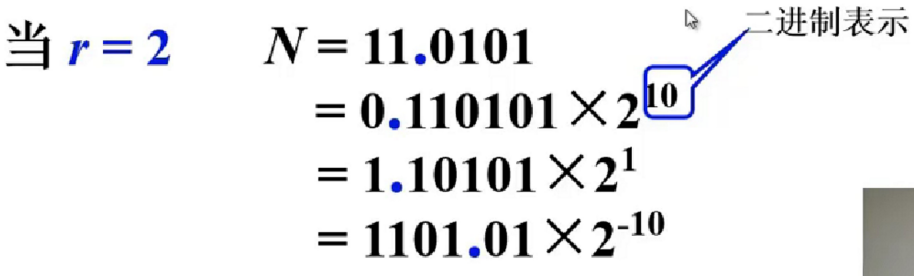
3、数据存储单元的利用率往往很低

浮点表示：

浮点数的一般形式：$N = S * r^j$ 和科学计数法类似

S尾数，j阶码（整数），r尾数的基值

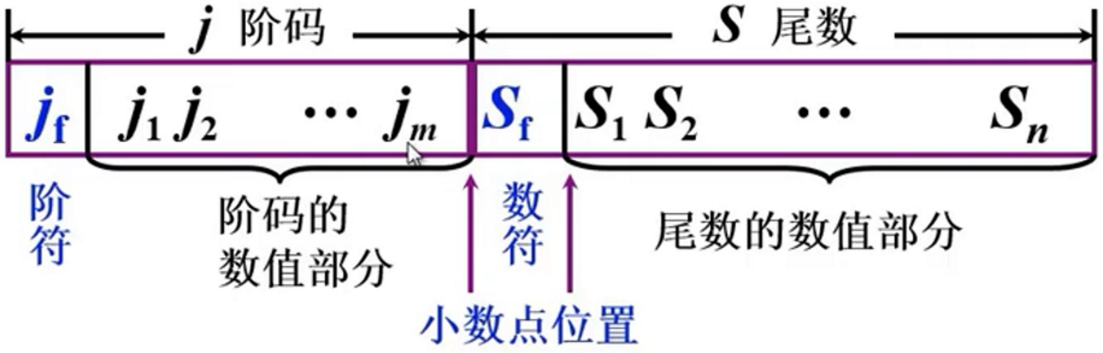
计算机中，r取2、4、8、16等



j=移动的位数，用二进制表示，小数点左移，j就是正的，移动2位就是10

S用小数表示，一般整数部分为0，可正可负

因此，在计算机中，浮点数具体的表示方法为



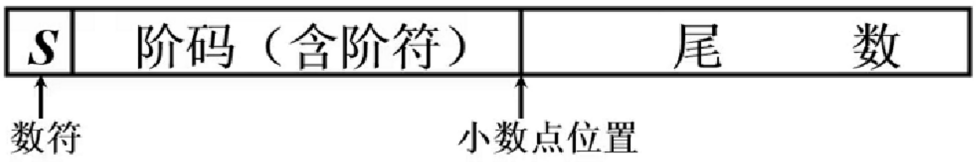
S_f 代表浮点数的符号

n 其位数反映浮点数的精度

m 其位数反映浮点数的表示范围

j_f 和 m 共同表示小数点的实际位置

但是注意，IEEE754标准是这样的



尾数为规格化表示

非“0”的有效位最高位为“1”（隐含）

机器零

当浮点数尾数为0时，均按机器0处理

当浮点数的阶码小于等于它所表示的最小数时，不论尾数为何值，均按机器0处理

定点运算

移位运算：小数点或数值进行移动，从而改变数据的大小

加减法运算：利用数的补码进行加减法运算，连通符号位一起加减，产生的进位直接丢掉

乘法运算：因为只有0和1，乘法要比十进制容易得多，基本就是一个数的浮点数移动然后相加

除法运算

浮点数四则运算

算术逻辑单元

ALU电路

快速进位链

进位链：传送进位的电路

串行进位链

进位串行传输

并行进位链

n位加法器的进位同时产生

指令系统

2022年6月18日 17:17

机器指令

指计算机cpu能直接识别和执行的指令

指令集：计算机软件和硬件的交界面，是所有机器指令的集合

指令的一般格式

操作码字段：反映机器做什么操作

1、长度固定（为了译码过程的方便），代表为RISC

2、长度可变，代表为X86，必须使用拓展操作码技术

地址码字段

四地址、三地址、二地址、一地址、零地址

指令的字长

取决于操作码字长、地址码字长、操作数地址的个数

若指令字长固定，则=存储字长

若不固定，则按字节的倍数变化

操作数类型

操作数：指令要进行处理的数据，包括：

地址，即无符号整数

数字，定点数、浮点数、十进制数

字符，用ascii码表示

逻辑数，逻辑运算

操作类型

1、数据传送

2、算术逻辑操作（包括算术和逻辑，也就是加减乘除等运算以及与或非等逻辑运算）

3、移位操作

4、转移指令

5、输入输出（对外部设备的数据）

寻址方式

寻址：确定本条操作的操作数地址或者下一条指令的指令地址

寻址方式：如何找到指令的地址或数据的地址

指令寻址：

顺序寻址

跳跃寻址

数据寻址方式：

形式地址：指令字中的地址

有效地址：操作数的真实地址

1、立即寻址

2、直接寻址：有效地址就是形式地址

3、隐含寻址：将地址隐藏在操作码中

4、简介寻址：有效地址由形式地址间接提供

5、寄存器寻址：有效地址即寄存器地址，数据保存在寄存器里

6、寄存器间接寻址：有效地址存在寄存器中

7、基址寻址

8、变址寻址

9、相对寻址

10、堆栈寻址

RISC精简指令集

80-20规律：典型程序中80%的语句只用到了不到20%的指令

执行频率高的简单指令，因复杂指令的存在，执行速度无法提高

RISC的特征：

选用使用频率高的简单指令，并使简单指令组合成复杂指令

指令长度固定、指令种类格式少、寻址方式少

只有load/store指令访存

CPU中有多个通用寄存器

采用流水技术，一个时钟周期内完成一条指令

采用组合逻辑实现控制器

时钟周期也称为振荡周期，定义为时钟频率的倒数。时钟周期是计算机中最基本的、最小的时间单位。在一个时钟周期内，CPU仅完成一个最基本的动作。时钟周期是一个时间的量。时钟周期表示了SDRAM所能运行的最高频率。更小的时钟周期就意味着更高的工作频率

CISC复杂指令集

CPU的结构和功能

2022年6月18日　23:21

CPU的功能

1、控制器的功能

取指令

分析指令

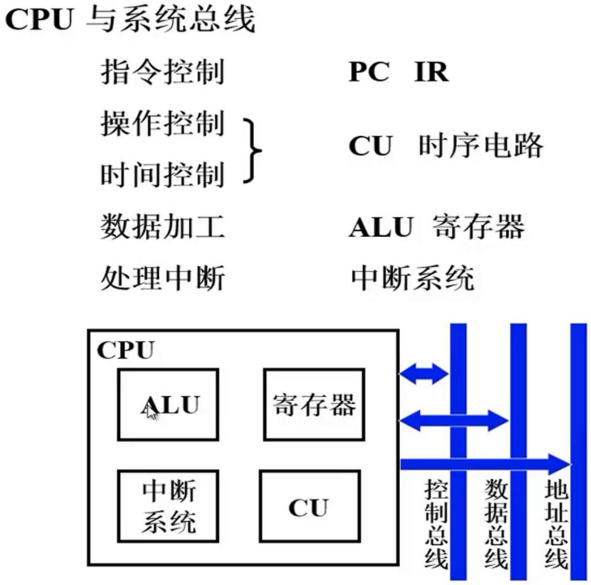
执行指令，发出各种操作命令

控制程序输入和结果的输出

总线管理

处理异常和特殊请求

2、运算器：实现算数和逻辑运算



指令周期

含义：取出并执行一条指令所需的总时长

分为取指周期和执行周期

每条指令的指令周期不同

提高计算机系统运行速度的方法：

1、提高各器件速度

2、改进系统结构，提升系统并发性

系统的并行性

并发：两个或以上的事件在同一时间段（时间片）内发生

同时：两个或以上的事件在同一时刻内发生，典型即流水线

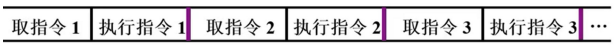
并行性的等级

指令级：指令之间、指令内部（细粒度，硬件实现）

过程级：程序、进程（粗粒度，软件实现）

指令的流水原理

1. 指令的串行执行



取指令　取指令部件　完成　总有一个部件 空闲
执行指令　执行指令部件　完成

2. 指令的二级流水

若 取指 和 执行 阶段时间上 完全重叠
指令周期 减半　速度提高 1 倍

实际上一条指令的完整执行过程有四个阶段，因此指令的流水不止二级，能达到三级甚至四级流水
但一般而言，速度往往不能提高几倍：

影响指令流水效率加倍的因素：

执行时间>取指令时间

解决办法：在取指令部件和执行指令部件之间增加指令部件缓冲区

条件转移指令（指令依赖）（必须等上一条指令执行完毕后才能知道下一条指令的地址）

解决办法：

解决办法：

猜测法

乱序执行（指令执行重排序，但执行完后必须按原指令顺序排序输出）：

例如：指令一正在执行还没出结果，指令二需要取值，目标为指令一的结果。但指令一还未执行完。这时观察到指令三和指令四没有指令依赖，因此考虑将其排序到指令二前面，优先执行，让总体执行效率更高

指令之间如果有依赖关系，叫做data dependency，而这件事发生在流水线相邻指令之间时，会产生data hazard，导致流水线停顿，或者流水线排空，CPU通过软硬件识别依赖配合调度策略，当然还有分支预测，来保证流水线不发生停顿，持续的执行指令

分支预测

分支预测是乱序执行的一个手段

我们在分支指令执行完之前，我们无法确定分支指令的下一条指令的地址，所以也就没法把分支指令的下一条命令放入流水线中，只能等待分支指令执行完毕才能开始下一条命令的取指步骤，所以流水线中就会出现气泡（Bubble）。气泡的产生会降低CPU运行效率，即无法满载。优秀的分支预测能消除大部分气泡

分支预测就是为了解决程序分支影响性能问题提出的，根据历史分支统计等，达到较高的准确度

如果预测成功，则继续执行；若预测不成功，则数据返回，重新执行

ROB重排序缓冲区

里面放着乱序执行前的指令的排列顺序，这个缓冲区越大，则一般代表这个处理器乱序执行程度越高，性能更加优越

影响指令流水线性能的相关因素

1、结构相关：不同指令争夺同一部件产生资源冲突

2、数据相关：不同指令可能存在对同一数据的重叠操作，可能改变数的读写访问等顺序

3、控制相关：由转移指令引起

流水线(Pipeline)技术，是指程序在执行时候多条指令重叠进行操作的一种准并行处理实现技术。通俗的讲将一个时序过程，分解成若干个子过程，每个过程都能有效的与其他子过程同时执行。这种思想最初是在RISC的架构中出现的，旨在提高处理器处理效率，争取在一个时钟周期中完成一条指令。

流水线性能

吞吐率：单位时间内流水线所完成的指令或输出结果的数量

加速比

效率：流水线中各功能段的利用率

超流水线技术：在一个时钟周期内再分段

中断系统

引起中断的因素：

人为中断

程序事故，如除法非法等

硬件故障

IO设备输入输出

除此之外，CPU内部还存在着解码器：

指令解码器：CPU内部有自己的指令编码，需要解码器翻译，运算器才能知道做什么

RISC和CISC处理器，目前最大的区别就在解码部分：CISC通过解码器将复杂指令输出类RISC的指令

RISC也需要解码器进行翻译，虽然接受的指令和执行单元执行的指令差别更小

解码器的效率如果不高，则就算后面运算器算力有余，也做不了事情

因此解码器的数量、解码器的效率能直接决定CPU的性能

每周期指令发射数：

这里的周期应该指指令周期

每个指令周期内解码器能发射出的最大指令数

指令发射后就给到了执行单元

数据冒险：

CPU的流水线设计，使原先有先后顺序的指令同时处理，当出现某些指令的组合时，可能会导致指令使用了错误的数据，这就是一种数据冒险

控制单元

2022年6月27日 15:45

完成一条指令的四个周期，总称指令周期：

取指周期

间址周期

执行周期

中断周期

控制单元的功能

发出各种控制命令或微指令，控制计算机各个部件，使其完成各种工作

多级时序系统

1、机器周期：所有指令执行过程中的一个基准时间

以完成最复杂指令功能的时间为基准

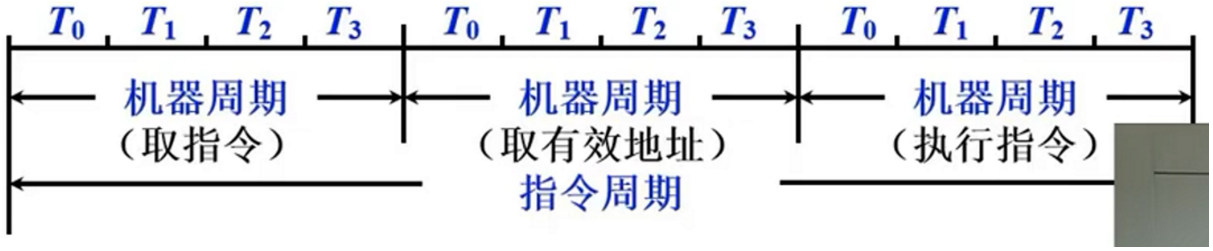
以访问一次存储器的时间为基准

2、时钟周期（节拍、状态）：是控制计算机操作的最小时间单位

一个机器周期包含数个时钟周期

3、指令周期

一个指令周期包含数个机器周期，一个机器周期包含数个时钟周期



控制方式

产生不同微操作命令序列所用的时序控制方式