**数的进制**

2转8

010 010 110

2 2 6

从右到左，不足左补0

8转2

除2取余，每个8为三个2，不足左补0

16转2同

补图片

原码：

+11 00001011

-11 10001011

使用原码直接计算会算出错误结果，所以原码的符号位不能直接参与计算，要和其他位分开

增加硬件的开销和复杂性

反码：

符号位不变，其他位取反

-11反码：11110100

反码符号位可以直接参与运算

补码：  
正数 原码补码反码相同

负数：反码+1

-11补码：11110101

由补码求原码即对次补码求补码

补码符号位参与运算，简化计算机电路，所以大部分使用补码

补码溢出：结果超范围

正+正 负+负

移码：补码符号位取反

直接使用移码补码计算结果正确

**定点数和浮点数**

区别：如何对待小数点

定点数：小数点隐含，固定在某个位置，若该位置在数的最低位之后就是定在整数

缺点：数值范围较大表示困难

浮点数：N=M\*R的e次方

M尾数，e指数，R基数

浮点数精度取决于M的宽度。范围取决于基数和指数

1000.01=1.00001\*2的三次方

=10.0001\*2的二次方

小数点在浮动，所以叫浮点数

机器指数固定不变所有指数可以不出现

运算：

对阶

对阶的原则是小阶对大阶，之所以这样做是因为若大阶对小阶，则尾数的数值部分的高位需移出，而小阶对大阶移出的是尾数的数值部分的低位，这样损失的精度更小

数据求复杂度

**校验码**：

奇偶检验：

海明码：

循环冗余校验码CRC：

**CPU**

运算器：

1. 加法器/累加器：专门存放算术或逻辑运算的一个操作数和运算结果的寄存器（运算器主要部分）
2. 程序状态寄存器：存放体现当前指令执行结果的各种状态信息（有无溢出）等，另一个存放控制信息（允许终端，跟踪）~

控制器

CPU核心，控制协调计算机动作

1. 程序计数器：存放下一条指令地址，顺序执行则自动+1，否则重填计数器
2. 指令寄存器：cpu将执行的操作码表
3. 指令译码器：将操作码解码告诉cpu做什么
4. 定时与控制电路：产生各种微操作控制信号
5. 堆栈和堆栈指针：一组寄存器或在存储器内的特定区域，因为寄存器数量有限，所有采用使用存储器软件堆栈；堆栈顶部的指针叫堆栈指针

**输入输出控制方式**

计算与外设之间数据交换叫输入/输出

1.程序查询方式

Cpu定时查询外设状态，若就绪则开始输入输出

串行查询：每次查一个

并行：将多个外设状态集中成一个专用端口，这样一次查询得到多个状态

最简单，但低效

2.中断方式

当cpu执行到I/O请求指令时，向输入输出控制器发出指令后，cpu不等待并继续执行其他任务。此时输入输出控制器负责和外设进行通信，当数据从数据寄存器写到外设或外设将数据写到数据寄存器后，输入输出向cpu发出中断请求，cpu响应并处理。由于数据寄存器大小有限，一次输入输出可能要经过多次中断才能完成

中断概念：打断cpu正在执行的工作，让cpu去处理其他任内

3.DMA方式（高效）

DMA直接存储器存取。无需cpu中转

Cpu处理到输入输出请求时，向DMA控制器发出相应指令，DMA若判断外设可用则填充地址寄存器，长度寄存器，向cpu发出总线请求信号，申请总线处理权，cpu让出，然后DMA控制器将数据在外设和内存制定区域间进行传送，而长度寄存器保存的值随着数据传输不断减少，为0时，向cpu发出中断请求，cpu响应中断，对内存数据进行后续处理

DMAC能访问系统总线和独立访问主存

（这两个特定使DMAC完成主存和输入输出设备之间数据交换）

DMA总线占有方式：

1. DMA传输时，cpu停止访问主存

优点：减少系统总线控制权交换次数，适合高速输入输出设备

缺点：cpu此过程无所事事

1. 时间片轮换：DMA和cpu总线权轮着给

缺点：效率低

1. 借用周期法：DMA操作时，DMAC控制总线访问内存，其他时间控制权在cpu

缺点：需要判定DMAC是否在用总线，复杂

适合外设速度慢的

DMA和中断区别

1. 中断方式，主存和输入输出控制器之间的数据传送需要cpu操作，还使用cpu的寄存器等
2. DMA拿到总线权，cpu暂停执行，现场不需要保护，数据传输完菜发送一个中断，通知cpu继续工作，没有切换任务操作
3. DMA并行性更好
4. 处理异常：中断强

**指令流和数据流**

指令流：机器执行的指令序列

数据流：由指令流调用的数据序列，包括输入数据和中间结果

SISD（单指令流单数据）：计算机每次处理一条指令，并只对一个操作部件分配数据

SIMD（单多数据）：并行处理机，具有多个处理单元，每次执行相同的指令，对不同的数据单元进行处理

MISD：多个处理单元，同时执行不同的指令，针对单一数据

MIMD：全面并行处理机（多处理机）

**流水线技术**

以硬件增加换取性能提升

把一个指令分解成更多小指令，由不同的处理单元来处理

锁存器：在个步骤间保存中间结果

负荷状态：所有工作单元都开始工作时，流水线才处于符合状态

理想状态：没有阻塞情况，某一环节出问题速度就会降低

影响流水线效率因素：

1. 条件转移指令

此情况下，下一条需要执行的指令未必是程序计数器所指定的指令，只有这条转移指令完成后才能判断下一条指令是什么

@1：若关闭流水线进入端口，防止错误发生会降低效率，条件转是移大量存在的

@2：猜测法：猜测可能会跳转到的语句，猜错则要清空流水线的内容

@3：必须执行的指令D提前执行，D执行完毕，条件转移结果出来，再判断B/C进入

流水线

1. 资源共享

若相邻的两条指令都对同一个资源操作，前写后读，在流水线时可能会出现错误，前一条指令没保存完，后一条指令的就开始读，读的是未改写的数据

解决：资源冲突时，暂停后读指令进入流水线，降速；编译时在冲突中间插入其他语句

以使进入流水线时间拉开

1. 寄存器

相邻的指令使用了相同的寄存器

给两个指令分配不同的寄存器

1. 中断系统

流水线遇到中断也会停止

流水线响应方式：

精确断点法：立即停止现有流水线，增加cpu硬件复杂度

不精确：中断时继续执行，停止流水线入口，流水线内执行执行后，再执行中断处理指

令；响应时间长但实现控制简单

**精简指令计算机**

指令系统：

指令系统是cpu所有指令的汇集

指令分解为 ：操作码+地址码

操作码：确定指令类型

地址码：确定指令所要处理的数据（根据地址码的个数0-n可以有四/三指令等）

指令长度相同：定长指令字结构，操控简单

长度不同且长度较长：变长指令字结构，容易扩展但增加硬件复杂度

类型：

1. 立即寻址

地址码就是操作数，不必再次访问内存取操作数，也无需修改操作数

1. 直接寻址

地址码是主内存的绝对地址，不必做任何换算

缺点：

@1：寻址范围有限，地址码的位数限制寻址空间

@2：随着内存越来越大，如果使用变长指令结构，指令会变得臃肿

1. 寄存器寻址

地址码的地址是寄存器地址。访问寄存器速度比内存寻址快，但寄存器数量有限

1. 间接寻址

地址码指向主存的数据，这个数据是一个地址，提高寻址灵活性，扩大寻址范围

但要多次读主存，速度低

1. 寄存器间接寻址

地址码保存的是寄存器地址，寄存器保存的是数据地址，快、灵活，使用广泛

CISC和RISC

图片

**存储器系统**

存储器中数据存取方式

顺序存储：数据访问按特定的线性顺序进行（磁带）

直接存取：与顺序存储类似，也使用一个共享读写装置对所有数据进行访问

但每个数据块都拥有唯一的地址标识，读写装置可以直接移动到目的数据块所在位置进行访问，存取时间也可变（磁盘）

随机存取：存储器的每一个可寻址单元都具有唯一的地址和读写装置，系统可以在相同的时间内对任意一个存储大暖的数据进行访问，与先前访问序列无关（主存）

相联存储：也是一种随机存取的形式，但选择某一单元进行读写取决于其内容而不是地址；

可以对所有存储单元的特定位进行比较，选择符合条件的单元进行访问（映射）

存储器性能衡量：存取时间，存储器带宽，存储器周期和数据传输量

主存（内存）

RAM：可以写入和读出，但断电信息无法保存，只能保存暂存数据

DRAM：动态随机存储，信息随时间逐渐消失，需要对其进行刷新，维持信息不丢失(主要)

SRAM：静态随机存储，不断电下信息一直保持

光盘存储器：

利用激光束在记录表面存储信息，根据激光束的反射光来读出信息

CD、DVD

存储量大、易于更换但速度慢

Cache（高速缓冲存储器）

提高CPU数据输入/输出速率

命中率：cpu需要数据能在缓存找到

Cache地址映射：cpu发出访问请求后，存储器地址先被送到Cache控制器以确定所需数据是否已经在Cache中，若命中则对Cache进行访问

特点：Cache产生一次未命中之后，相应数据同时读入cpu和Cache；当cache存满数据后会淘汰旧数据

保证cache数据和主存内容一致

写直达（写通）：当要写cache时，数据同时写回主存

写回：cpu修改cache一行后，相应的数据并不立即写入主存，而是当该行从cache被淘汰时菜把数据写回主存

标记法：对cache的每一个数据设置一个有效位。当数据进入cache后，有效位为1；当cpu对该数据修改，数据只写入主存并将该有效位清0；当要从cache读取数据时需要测试其有效位，1直接从cache取，否则从主存取

RAID（廉价磁盘冗余阵列）

目的：缩小cpu和磁盘速度差距

方法：用多个较小的驱动器替换单一的大容量磁盘驱动器，同时在多个磁盘上分布存放数据

以支持同时从多个磁盘进行读写，从而改善I/O性能

特点：

多个磁头同时操作增加出错概率

使用存储校验信息来从错误中恢复数据