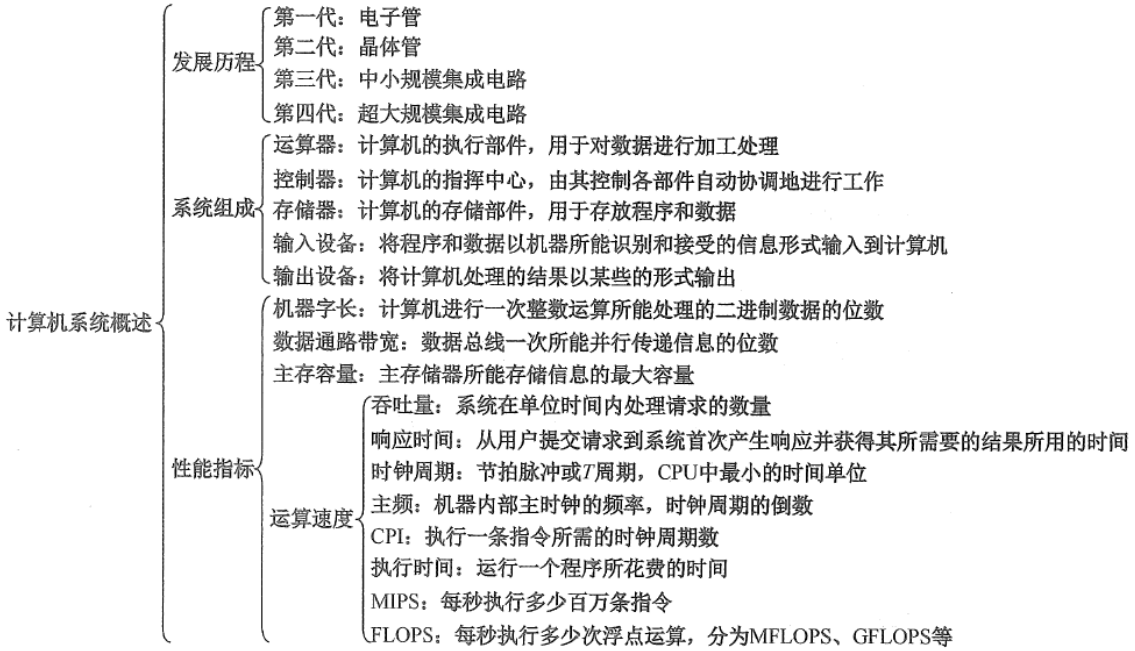
计算机组成原理

第一章计算机系统概论



1.1 计算机系统组成

计算机系统是由“**硬件**”和“**软件**”组成。

1）计算机系统由硬件和软件两部分**组成**。
2）计算机系统**性能**由硬件和软件共同决定。

1.2 计算机系统5层层次结构

1）（从下到上）微程序机器、传统机器、操作系统机器、汇编语言机器、高级语言机器
2）微程序机器和传统机器是**物理机**，其他是**虚拟机**。

1.3 编译与解释

将高级语言直接翻译成机器语言的程序软件叫**翻译程序**

1）**编译**：将一种语言编写的程序全部翻译成另一种语言，然后再执行；
2）**解释**：将一种语言编写的程序的一条语句翻译成另一种语言的一条或多条语句，然后执行，执行完这条语言后，再解释下一条。

1.4 计算机体系结构 | 计算机组成

1）计算机体系结构是指那些能够被**程序员**看到的计算机的属性。如指令集、数据类型等；
2）计算机组成是指如何**实现**计算机体系结构所体现出来的属性；
3）以**乘法指令**为例，计算机是否有乘法指令，属于体系结构的问题。乘法指令是采用专用的乘法器，还是使用加法器和移位器构成，属于计算机组成的问题。

1.5 冯诺依曼机器的主要特点

1）计算机由**运算器、存储器、控制器、输入设备**和**输出设备**五大部分组成；
2）指令和数据存储在**存储器**中，并可以**按地址访问**；
3）指令和数据均以**二进制**表示；
4）指令由**操作码和地址码**构成，操作码指明操作的性质，地址码表示操作数在存储器中的位置；
5）指令在存储器内按**顺序存放**，通常按自动的顺序取出执行；
6）机器以**运算器**为中心，I/O设备与存储器交换数据也要通过运算器。（因此，现代计算机以存储器为中心的计算机结构）

> “存储程序”的概念是指将指令以代码的形式事先输入计算机的主存储器，然后按其在存储器中的首地址执行程序的第一条指令，以后就按该程序的规定顺序执行其他指令，直至程序执行结束。

1.6 存储单元、存储字、存储字长、存储体

存储单元：存储一个存储字并具有特定存储地址的存储单位；
存储字：一个存储单元中存放的二进制数据
存储字长：存储字中二进制数据的位数
存储体：由多个存储单元构成的存储器件。

1.7 主存储器中，什么是MAR，什么是MDR，存储器的最大容量由什么决定?

1）**MAR**：存储地址寄存器，保存需要访问的存储单元地址。反映存储单元的个数。
2）**MDR**：存储数据寄存器，缓存读出/写入存储单元的数据。反映存储字长。
3）存储器的**最大容量**由MAR寄存器的位数和MDR寄存器的位数决定。

1.8 机器字长，存储字长

机器字长：CPU一次能够处理的二进制数据的位数。
存储字长：按照某个地址访问某个存储单元获取的二进制数据的位数。

1.9 假设MAR寄存器的位数为16位，MDR寄存器的位数为16位，存储器的最大容量是多少?

1）MAR寄存器的位数为16位，能表示的地址个数为2的16次方，为64K；
2）MDR寄存器的位数为16位，说明存储字长为16位，也即2个字节；
3）存储器的最大容量为64K * 2B = 128K Byte

1.10 计算机的工作过程

1. 把程序和数据装入主存储器。
2. 将源程序转换成可执行文件。
3. 从可执行文件的首地址开始逐条执行指令。

1.11 指令执行的过程

（以取数为例）

取指令： PC —>MAR—>M—>MDR—>IR
根据PC 取指令到IR, 将PC 的内容送MAR, MAR 中的内容直接送地址线，同时控制器将读信号送读 / 写信号线，主存根据地址线上的地址和读信号，从指定存储单元读出指令，送到数据线上， MDR 从数据线接收指令信息，并传送到IR 中。

分析指令： OP(IR)—>CU
指令译码并送出控制信号。控制器根据IR 中指令的操作码，生成相应的控制信号，送到不同的执行部件。在本例中, IR 中是取数指令，因此读控制信号被送到总线的控制线上。

执行指令: Ad(IR)—>MAR—>M—>MDR—>ACC
取数操作。将IR 中指令的地址码送MAR, MAR 中的内容送地址线，同时控制器将读信号送读 / 写信号线从主存指定存储单元读出操作数，并通过数据线送至MDR, 再传送到ACC 中。

此外，每取完一条指令，还须为取下一条指令做准备，形成下一条指令的地址，即(PC)+1 —> PC 。

1.1 2计算机的主要性能指标

机器字长：ＣＰＵ一次能处理的二进制数据的位数。

数据通路带宽：数据总线一次所能并行传送信息的位数。

主存容量：主存储器所能存储信息的最大容量

运算速度

(1) 吞吐量和响应时间。

吞吐量：指系统在单位时间内处理请求的数量。

响应时间：指从用户向计算机发送一个请求，到系统对该请求做出响应并获得所需结果的等待时间。

(2) 主频和CPU 时钟周期。

主频：机器内部时钟的频率。

CPU 时钟周期：主频的倒数，它是CPU 中最小的时间单位，每个动作至少需要1 个时钟周期。

(3)CPI(Clock cycle Per Instruction),即执行一条指令所需的时钟周期数。

第二章中央处理器

2.1 ＣＰＵ的功能

中央处理器(CPU) 由**运算器和控制器**组成。CPU 的具体功能包括:

指令控制。完成取指令、分析指令和执行指令的操作，即程序的顺序控制。
操作控制。一条指令的功能往往由若干操作信号的组合来实现。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。

2.2 流水线越多，并行度就越高。是否流水段越多，指令执行越快?

错误，原因如下:

流水段缓冲之间的**额外开销增大。**每个流水段有一些额外开销用于**缓冲间传送数据、进行各种准备和发送等功能**，这些开销加长了一条指令的整个执行时间，当指令间逻辑上相互依赖时，开销更大。

流水段间**控制逻辑变多、变复杂。**用于**流水线优化**和存储器（或寄存器）**冲突处理**的控制逻辑将随流水段的增加而大增，这可能导致用于流水段之间控制的逻辑比段本身的控制逻辑更复杂。

第三章系统总线

3.1 引入总线目的

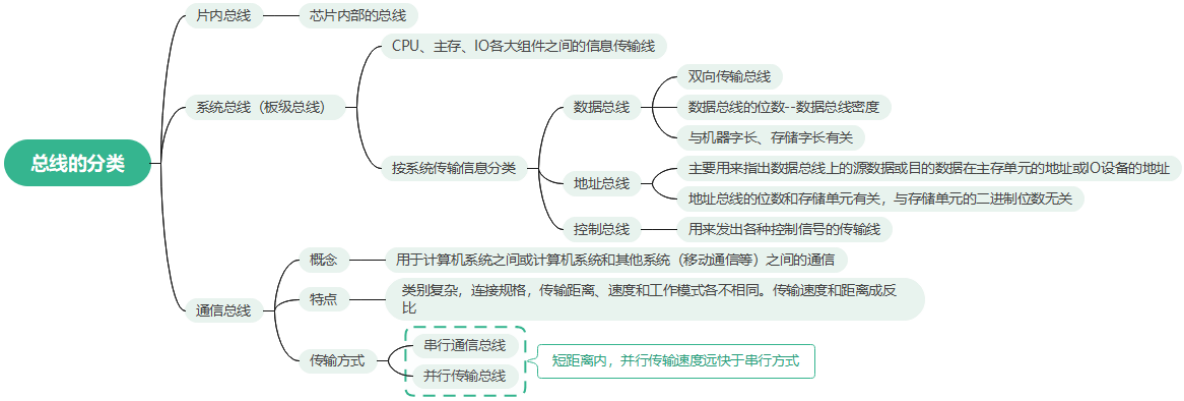
在冯诺依曼结构中，各个部件之间均有单独连线，不仅线多，而且导致扩展I/O设备很不容易。即扩展一个I/O设备，需要连接很多线。
因此，引入了总线连接方式，将多个设备连接在同一组总线上，构成设备之间的公共传输通道。

3.2 总线的两大基本特征是什么?

1）**共享**：多个部件连接在同一组总线上，各个部件之间都通过该总线进行数据交换。
2）**分时**：同一时刻，总线上只能传输一个部件发送的信息；

3.2.1 总线的特性

3.3 总线的分类



3.3.1 系统总线按照传输信息的不同，分成哪几类？是单向的，还是双向的？

1）分成数据总线、地址总线以及控制总线。
2）**数据总线**：各个功能部件之间传送数据信息，**双向**传输；
3）**地址总线**：用来指明数据总线上，源数据或目的数据所在的主存单元的地址。**单向**：由CPU发出
4）**控制总线**：用来发送各种控制信号。对于控制总线中的单根线，是单向的，即只能由一个部件发向另一个部件。而一组控制总线中，有输入也有输出，因此，控制总线也可以看成是双向的。

3.4 总线宽度、总线带宽、总线复用、信号线数

1）**总线宽度**：数据总线的根数，一般是8的倍数。是衡量计算机系统性能的重要指标；
2）**总线带宽**：即总线数据传输速率，总线上每秒能够传输的最大字节量。
3）**总线复用**：一条信号线上分时传送两种信号。例如数据总线和地址总线的分时复用；
4）**信号线数**：地址总线、数据总线和控制总线三种总线的线数之和。

3.5 假设总线的工作频率为33MHz，总线宽度为32位，则它最大的传输速率是多少？

33 * （32/8） = 132 MB/s

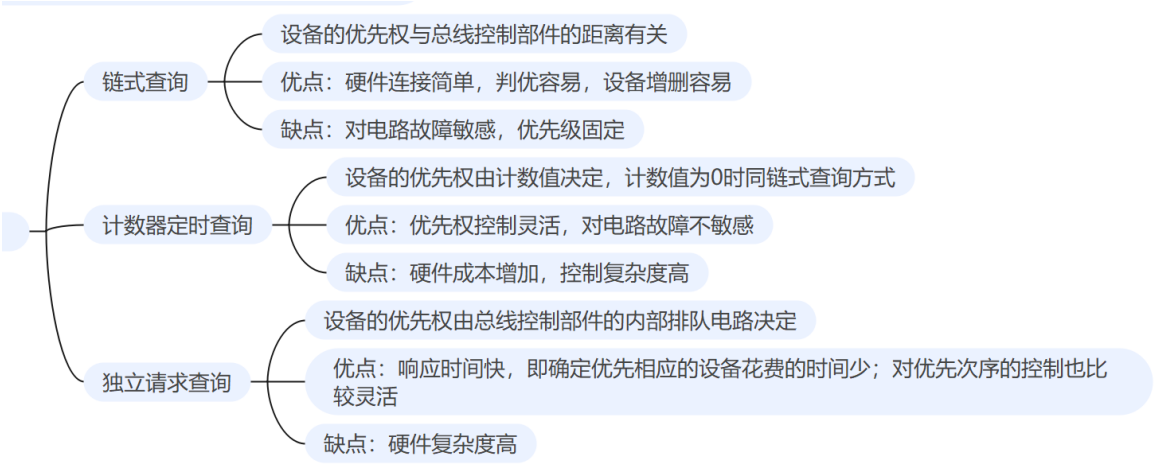
3.6 单总线结构的概念及缺点（现代计算机为什么要采用多总线结构？）

在**单总线结构**中，所有的部件（CPU、主存、I/O设备）都连接在一组总线上。
但所有的信息传送都要通过这组总线，同时只能有一个部件向总线上发送信息，导致总线成为系统的瓶颈。
因此，发展出来了**多总线结构**，其基本思想均是将速度相近的设备挂接在同一组总线上，总线之间通过总线控制器相连。

3.7 集中式总线判优控制有哪三种方式，哪种方式的优先级不能改变

1）链式查询、计数器定时查询、以及独立请求。
2）链式查询的优先级不能改变，离控制器最近的优先级最高。

3.8 简述链式查询、计数器定时查询以及独立请求三种方式的工作原理。



3.9 总线周期及其阶段?

1）**总线周期**：总线上两个部件完成一次完整且可靠的**数据传输**时间；
2）分为四个阶段：
申请分配阶段：申请总线
寻址阶段：发出地址及有关命令
传数阶段：进行数据交换
结束：从总线上撤除信号，让出总线

3.10 总线通信控制概念及分类

1）**总线通信控制**：解决通信双方如何获知传输开始和传输结束，以及如何协调配合；
2）**同步**通信、**异步**通信、**半同步**通信、**分离式**通信

3.10.1 同步通信

1）**同步通信**：总线上各个部件由统一的时钟信号控制；在总线周期中，每个时钟周期各个部件如何动作都有明确的规定。
2）**优点**：规定明确，模块间配合简单一致
3）**缺点**：强制性同步，速度不同的部件，严重影响工作效率，不灵活

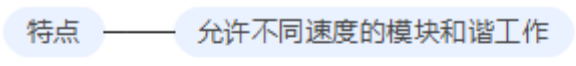
4）**适用场合**：短距离，各部件存取时间一致

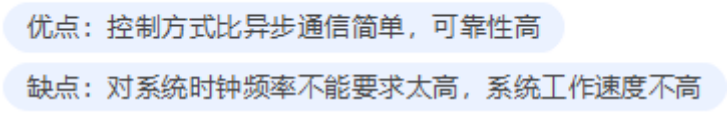
3.10.2 异步通信

1）**异步通信**：总线上各部件没有统一的时钟标准，采用**应答式**通信；（主模块发出请求后，一直等到从模块反馈回来应答信号之后才开始通信）
2）不互锁、半互锁、全互锁。

3.10.3 半同步通信

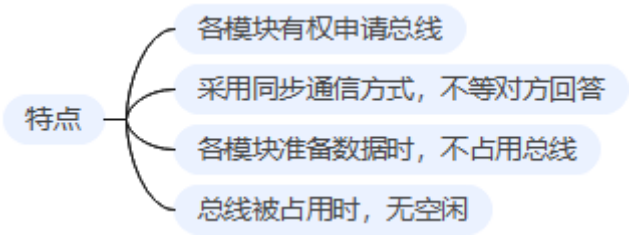
1）半同步通信**结合同步通信和异步通信。**
同步通信：采用统一的时钟，规定了在一定的时钟周期干什么事情；
异步通信：如果从模块没有准备好，增加一个“等待响应”信号。



2）


3.10.4 分离式通信

主模块发出地址和命令之后，放弃总线，在从模块准备数据期间，使得总线可以被其他设备所用。提高总线利用率。
但是，这种方式控制比较复杂。



3.11 波特率、比特率

波特率：单位时间内传送的二进制数据的位数，单位bps
比特率：单位时间内传送的有效的二进制位数。

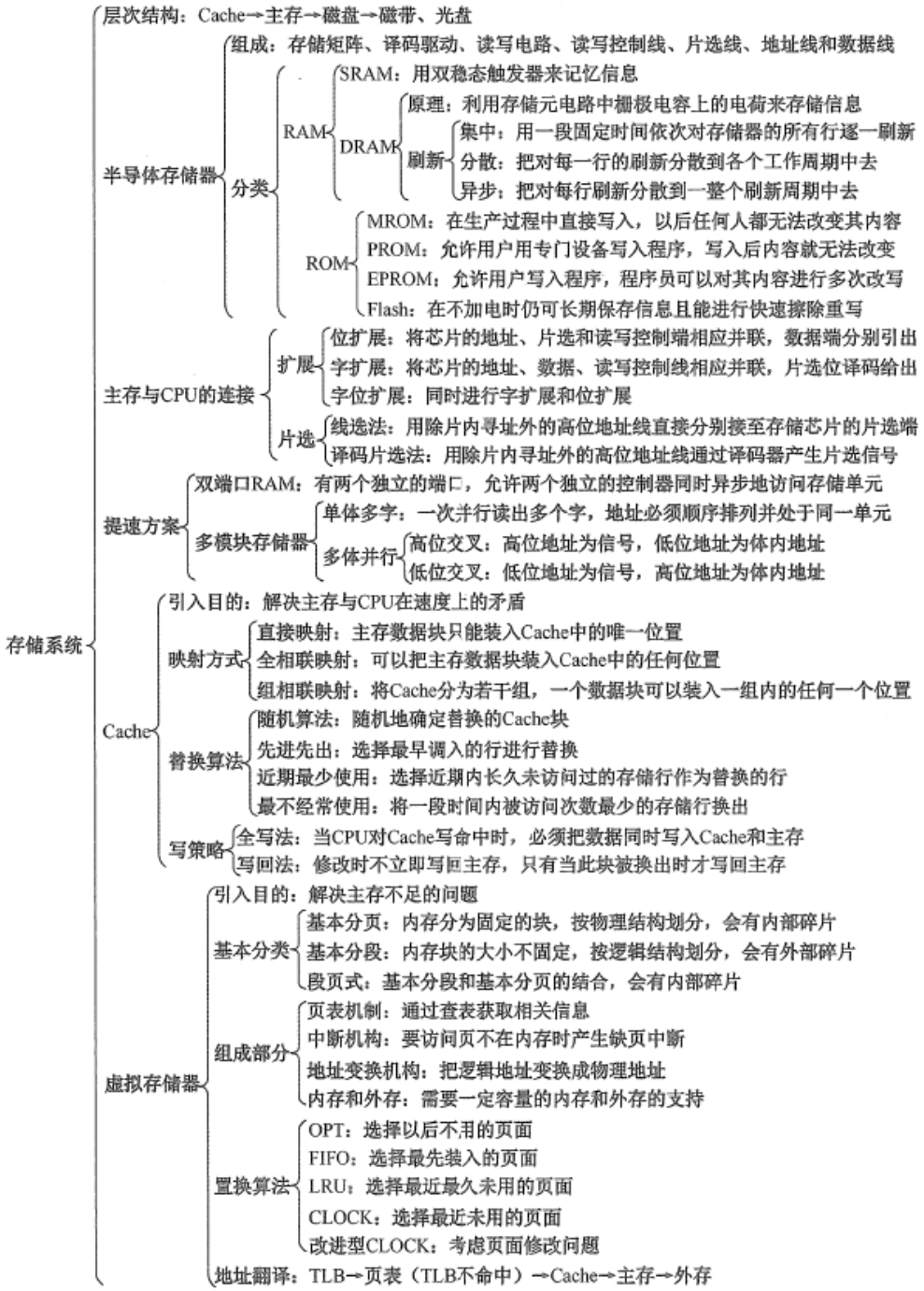
3.12 异步通信时，常规需要设置的参数有哪些?

波特率、停止位（1/2/1.5）、校验位（奇校验、偶校验、无校验）

3.13 汉明码、奇偶校验码

1）**奇偶校验**只能检错，不能纠错。
2）**汉明码**可以纠错

第四章存储器



4.1 存储器按存取方式，可以分成哪四类？哪些属于随机访问存储器，哪些属于串行访问存储器？

1）可以分为**随机**存储器、**只读**存储器、**顺序**存储器和**直接**存储器；
2）随机存储器和只读存储器属于随机存储器，即**存取时间与物理地址无关**；
3）顺序存储器（典型的如磁带）和直接存储器（典型的如磁盘）属于串行存储器，即**存取时间与物理地址有关。**

4.2 衡量存储器使用哪三个指标？寄存器、缓存、主存中，哪个速度最快？哪个最便宜

1）速度、容量、价格。
2）寄存器速度最快，主存最便宜。

4.3 常见的存储系统层次结构有哪两种？透明性如何？各自用来解决什么问题的？

1）**缓存-主存层次**：用来缓解**CPU和主存速度不匹配**的问题，由**硬件**来完成，对所有的**程序员**完全透明。
2）**主存-辅存层次**：用来解决**主存容量不够**的问题，由**操作系统和硬件**共同完成，对**应用**程序设计者透明，对**系统**程序设计者不透明。

（现在一般存储器都即能按字访问，也能按照字节访问，因此，存储器编址时，每个字节都有一个独立的地址。）

4.4 字在存储单元中有两种存储方式，大端方式和小端方式。各是什么含义？x86采用的是哪种存储方式?

1）**大端方式**：字的低位存在内存的高地址中，而字的高位存在内存的低地址中；
2）**小端方式**：字的低位存在内存的低地址中，而字的高位存在内存的高地址中。
3）**x86**CPU采用的是**小端**方式。

4.5 主存的三个主要技术指标

存储容量、存取速度和存储带宽

4.6 什么是存取时间？什么是存取周期？哪个大?

1）**存取时间**：启动一次存储器完成本次操作（读或写）所需的时间；
2）**存取周期**：连续两次启动存储器所需要的最小间隔时间；
3）存取周期**包含**存取时间；

4.7 什么是存储器带宽?

单位时间内存储器存取的信息量；

4.8 半导体存储芯片译码驱动包含哪两种方式，请简要说明。

1）**线选法**：**所有**的地址芯片通过**一个译码器译码**，选择一个存储单元的各位，适合于**存储容量不大**的芯片；
2）**重合法**：将地址分为两组，**每组**通过**一个译码器译码**，选择行或列，**行、列交叉处**就是要访问的存储位。

4.9 随机存储器包含哪两大类？哪个需要刷新？请从速度、容量、价格等方面进行简要比较。

主存储器由DRAM 实现，靠处理器的那一层(Cache) 则由SRAM 实现，它们都属于**易失性存储器**，只要电源被切断，原来保存的信息便会丢失。DRAM 的成本低于SRAM, 速度也慢于SRAM。而ROM 属于非易失性存储器。

1）**静态RAM**：采用**触发器**原理实现；

> 通常把**存放一个二进制位**的物理器件称为**存储元**，它是存储器的最基本的构件。地址码相同时**多个存储元**构成一个**存储单元**。若干**存储单元**的集合构成**存储体**。
>
> 静态随机存储器(SRAM) 的存储元是用**双稳态触发器**（六晶体管MOS) 来记忆信息的，因此即使信息被读出后，它仍保持其原状态而不需要再生（非**破坏性读出**）。SRAM 的**存取速度快**，但**集成度低**，**功耗较大**，所以一般用来组成**高速缓冲存储器**。

2）**动态RAM**：采用**电容**原理实现，需要**刷新**。

> 动态随机存储器(DRAM) 是利用存储元电路中**栅极电容上的电荷**来存储信息的。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。相对于SRAM 来说， DRAM 具有容易集成、位价低、容量大和功耗低等优点，但DRAM 的存取速度比SRAM 的慢，一般用来组成大容量主存系统。DRAM 电容上的电荷一般只能维持1~2ms, 因此即使电源不断电，信息也会自动消失。为此，每隔一定时间必须**刷新**，通常取2ms, 这个时间称为刷新周期。常用的刷新方式有3 种：集中刷新、分散刷新和异步刷新。

3）相比于动态RAM，静态RAM的速度快、容量小、价格高，一般用于**缓存**，而动态RAM一般用于**内存**。

4.10 只读存储器有哪几种?

1）掩模ROM（MROM）：出厂后内容不能被更改。
2）PROM：可编程只读存储器，可以进行一次性编程；
3）EPROM：可擦除只读ROM，用紫外线照射；
4）EEPROM：电可擦除只读ROM。
6）FLash Memory：采用EEPROM的非易失性存储器。

特点

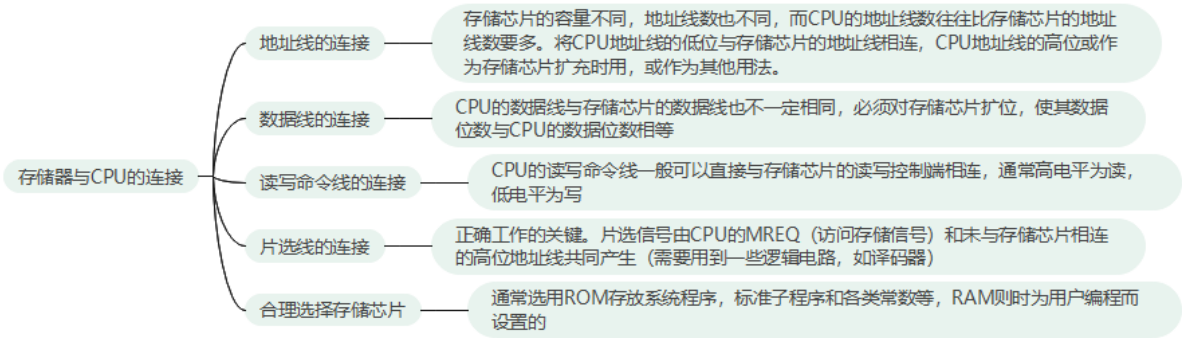
1. 结构简单。
2. 具有非易失性。

4.11 存储器的扩展

存储器的扩展通常有位扩展和字扩展

1）**位扩展**：增加存储器的字长，例如两个1K * 4位的存储芯片构成1个1K*8位的存储器；
2）**字扩展**：增加存储器的字数，例如两个1K * 8位的存储芯片构成1个2K * 8位的存储器；
通常字扩展和位扩展两种方式混合使用。

4.12熟虑掌握存储器的扩展，包括地址空间分配、地址线的连接、数据线的连接、片选信号的产生及连接等



4.13 假设欲检测的二进制代码为n位，为了使其具有1位的纠错能力，需添加K位检测位，组成n+k位的代码。问，应添加多少位检测位?

应添加的检测位位数：2的k次方大于等于n+k+1。
因为要使其有1位的检测能力，必须使用k位来说明n+k位到底哪一位出现了错误，k位能表达的数量为2的k次方，而n+k位到底哪一位
出现了错误或者是全部正确，共有n+k+1种状况，因此，k的取值需要满足：2的k次方大于等于n+k+1

4.14 对于汉明码，应熟练掌握汉明码的编码方式（按照配偶或配奇的原则），以及给出汉明码，得到要传送的原始信息（包括纠错过程）

4.15 提高访存速度的三种方式

1）采用**高速元器件**；
2）采用**存储层次**结构：cache-主存结构；
3）调整**主存结构**：包括**单体多字，多体并行**两种方式。

4.16 单体多字存储器

1）单体多字存储系统**一次访存取出多个字**，地址必须顺序排列并处于同一存储单元。存储器中只有一个存储体。
2）优点是：显著提高了存储器带宽。

4.17 多体并行存储器

由多体模块组成。每个模块都有相同的容量和存取速度，各模块都有独立的读写控制电路、地址寄存器和数据寄存器。

1）**高位交叉编址方式**：存储体的编址方式为**顺序存储**，即一个存储体存满后，再存入下一个；存储单元地址的高位为存储体的编号。
高位交叉编址并不能提高单次访存速度，但能使**多应用并行访存**，提高系统的并发性。

优点：各存储体可并行工作，体内地址连续，便于存储器的扩充

缺点：由于数据和程序按顺序存放，导致某个存储体访问过于频繁，其余的存储体空闲

特点：多体并行提高访问速度，有利于存储容量的扩展；连续读取n个字所需时间为nT（T为存储周期）

2）**低位交叉编址方式**：存储体的编址方式为**交叉存储**。即程序连续存放在相邻的存储体之中。存储单元地址的低位为存储体的编号。
低位交叉编址能显著提高单次访存速度。

优点：充分挖掘总线的每个瞬间（分离式通信），结合流水技术，有利于增加存储器的带宽。

特点：在不改变单体的存储周期的前提下。结合流水线技术增加存储器的带宽。连续取n个字时间T+（n-1）τ，τ总线传输周期

4.18 在四位低位交叉编址中，假设存取周期为T，总线传输周期为τ，为了实现流水线方式存储，应满足什么条件？如果连续读取四个字，所需要的时间是多少？

1）T= 4τ
2）连续读取四个字，所需要的时间为T + （4-1）τ
注意：假设不是低位交叉编址，而是高位交叉编址，连续读取四个字所需要的时间仍然为4T。

4.19 在CPU和内存之间引入cache的原因

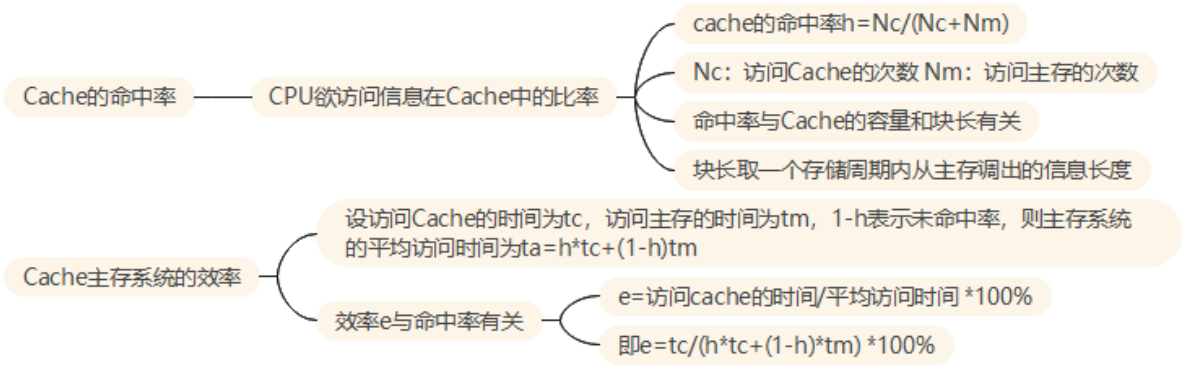
1）避免CPU空等I/O访存；
2）缓解CPU和主存速度不匹配的问题。

> Cache存储器：电脑中为高速缓冲存储器，是位于CPU和主存储器DRAM（Dynamic Random Access Memory）之间，规模较小，但速度很高的存储器，通常由SRAM（Static Random Access Memory静态存储器）组成。

4.20 程序的局部性原理

CPU从主存取指令或数据，在一定时间内，只是对主存局部地址区域访问

4.21 Cache命中率、平均访问时间以及访问效率的计算

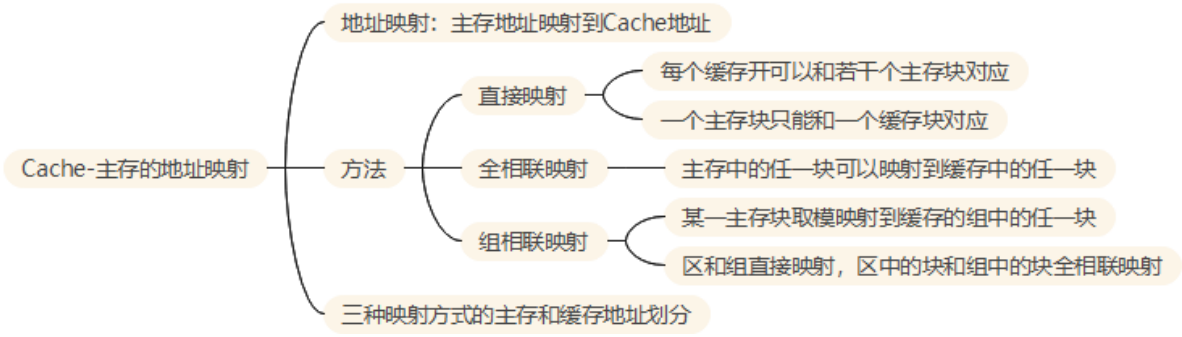


4.22 Cache写操作有哪两种方式

1）**写直达法**：写操作既写入Cache又写入主存；
2）**写回法**：只把数据写入Cache而不写入主存，当Cache中数据被替换出去之后才写入主存。

3）**标记法**

4.23 将主存地址映射到Cache地址称为地址映射，常见的Cache映射方式有哪几种



4.24 需要大家掌握各种映射方式之下，写出主存地址格式、cache地址格式，以及主存地址向cache地址的转换。

4.25 Cache常用的替换算法有哪些？哪个命中率最高？

当Cache产生了一次访问未命中之后，相应的数据应同时读入CPU和Cache。但是当Cache已存满数据后，新数据必须替换（淘汰）Cache中的某些旧数据。

1）先进先出、近期最少使用算法和随机替换算法；
2）命中率最高的是近期最少使用算法；

4.26 磁盘的三地址结构包括哪些？

柱面、磁头号和扇区号

第五章输入输出系统

5.1 I/O系统的发展大致可以分为哪4个阶段

1）早期（分散连接、串行工作、程序查询）
2）接口模块和DMA阶段（总线连接、并行工作、中断及DMA）
3）通道阶段（通道是具有特殊功能的处理器）
4）I/O处理机阶段
I/O系统的发展实际上是逐步将CPU从繁重的I/O工作中解放出来的过程；

5.2 I/O设备编址有哪两种方式？各有什么优缺点？

1）**统一编址方式**：和**存储器统一编址**，I/O地址作为存储器地址的一部分；无须用专用的I/O指令，但占用存储器空间。
2）**独立编址方式**：和**存储地址分开编**址，需用**专用的I/O指令。**

5.3 I/O设备与主机的联络方式有哪几种?

I/O设备与主机间交互信息时必须了解彼此的状态。根据I/O设备工作速度的不同，可以分为3类：
1）**立即响应**：不管其状态（认为其时刻准备好），适用于慢速设备。
2）**应答信号**：通过应答信号来进行交互；
3）**同步时标**：采用统一的时钟信号。

5.4 I/O总线包括哪四类?

数据线、设备选择线、状态线、命令线

5.5 I/O设备通常使用D触发器（完成触发器）和B触发器（工作触发器）来标识设备所处的状态。

D=0，B=0：暂停状态
D=0，B=1：准备状态
D=1，B=0：就绪状态

5.6 程序查询的基本工作原理

CPU不断去查询I/O设备状态，导致CPU和I/O设备串行工作。

5.7 中断的概念

计算机在执行程序过程中，当出现**异常清空或特殊请求**时，计算机停止现行程序的运行，转去处理这些异常清空或特殊请求，处理结束后，再返回现行程序的间断处，继续执行原程序，即为中断。

5.8 中断服务程序的基本流程包括哪四部分?

1）保护现场
2）中断服务
3）恢复现场
4）中断返回

5.9 什么是单重中断和多重中断?

1）**单重中断**：不允许中断现行的中断服务程序；
2）**多重中断**：允许级别更高的中断源中断现行的中断服务程序，也称为中断嵌套；

5.10 CPU响应中断的时机?

当前指令执行完毕后，cpu发出中断查询信号，也就是说，中断响应一定是在每条指令执行结束之后进行的，不可能在指令执行过程中响应中断。

ＣＰＵ响应中断具备的条件

1)在**CPU 内部**设置的**中断屏蔽触发器**必须是**开放**的。

2)**外设有中断请求**时，**中断请求触发器**必须处于**"1" 状态**，保持**中断请求信号。**

3)**外设**（接口）**中断允许触发器**必须为"**1**"'这样才能把**外设中断请求送至CPU 。**

具备上述三个条件时， CPU 在**现行指令结束的最后一个状态周期**响应中断。

５.１１中断响应优先级和中断处理优先级

中断响应优先级是由**硬件排队线路**或**中断查询程序**的查询顺序决定的，不可动态改变。

中断处理优先级可以由**中断屏蔽字**来改变，反映的是正在处理的中断是否比新发生的中断的处理优先级低（屏蔽位为"0", 对新中断开放），若是，则中止正在处理的中断，转到新中断去处理，处理完后再回到刚才被中止的中断继续处理。

5.12 向量中断、中断向量、向量地址

1)中断向量：每个**中断源**都有对应的处理程序，这个处理程序称为**中断服务程序**，其入口地址称为**中断向量。**所有中断的中断服务程序入口地址构成一个表，称为**中断向量表**；也有的机器把中断服务程序**入口的跳转指令**构成一张表，称为**中断向量跳转表。**

2)向量地址：中断向量表或中断向量跳转表中每个**表项**所在的**内存地址**或表项的**索引值**，称为**向量地址或中断类型号。**

3)向量中断：指一种**识别中断源**的技术或方式。识别中断源的目的是找到中断源对应的中断服务程序的入口地址的地址，即**获得向量地址。**

5.13 什么是DMA?

DMA：直接内存访问。在**主存和I/O设备**之间建立独立的总线连接。

5.14 在DMA方式中，由于DMA接口与CPU共享主存，可能会出现两者争用主存的冲突，为解决冲突，DMA和主存交换数据时，通常采用哪三种工作方式?

1）**停止CPU访问主存**：DMA访存优先级高；
2）**周期挪用（窃取）**：DMA挪用存储或窃取总线使用权一个或几个主存存取周期；
3）**DMA和CPU交替访问**：将CPU工作周期分成两部分，一部分供DMA访存，一部分供CPU访存。

5.15 DMA工作过程包括哪三部分?

DMA请求、DMA响应、数据传输、DMA结束

1）预处理
2）数据传输
2）后处理

5.16 程序中断和调用子程序的区别

两者的根本区别主要表现在**服务时间和服务对象**上不一样。

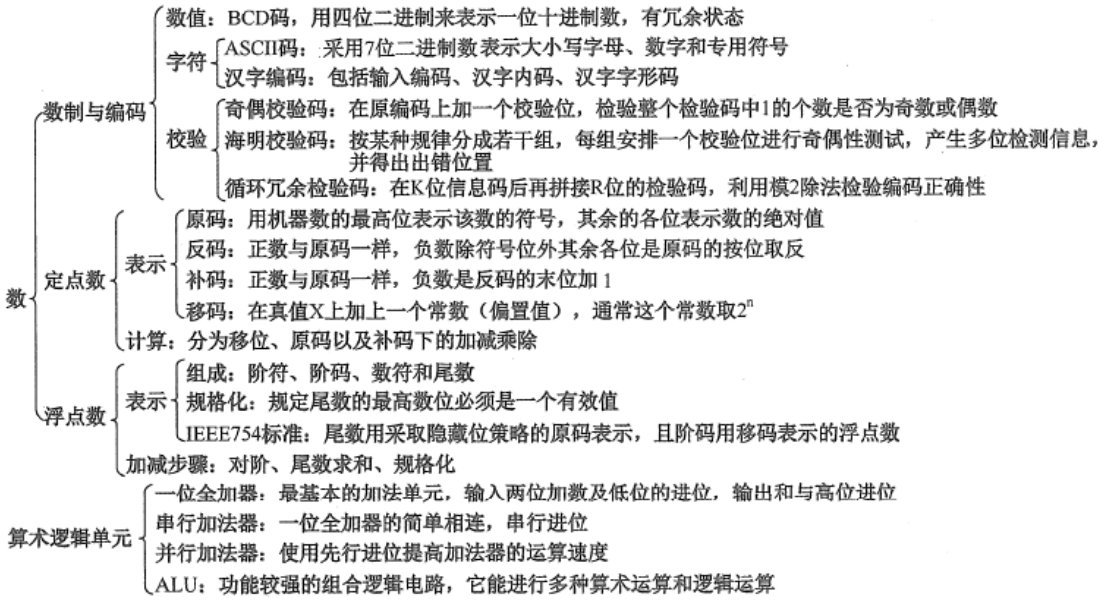
调用子程序过程发生的时间是**已知的和固定的。**而**中断过程发生**的时间一般是**随机**的。也可以说，调用子程序是**程序设计者事先安排**的，而执行中断服务程序是由**系统工作环境随机决定**的。

子程序完全为主程序服务，两者属于**主从关系。**而中断服务程序与主程序二者一般是无关的，，两者是**平行关系。**

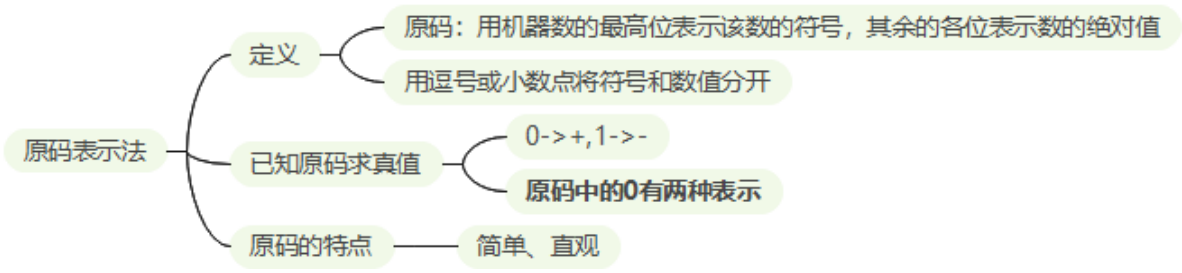
主程序调用子程序的过程完全属于**软件处理过程**，不需要专门的硬件电路；而中断处理系统是一个**软/硬件结合的系统**，需要专门的硬件电路才能完成中断处理的过程。

子程序嵌套可实现若干级，嵌套的最多级数受**计算机内存开辟的堆栈大小**限制；而中断嵌套级数主要由**中断优先级**来决定，一般优先级数不会很大。

第六章计算机的运算方法



6.1 有符号数的原码计算方法，以及通过原码求真值



6.2 补码计算的方法，通过补码求原码，然后求真值的方法

1）通过原码求补码：符号位不变，各位取反，末位加1；
2）通过补码求原码：符号位不变，各位取反，末位加1；

6.3 原码中0有2种表示方法（正零和负零），补码中0只有一种表示方法（正零和负零的表示方法一致）

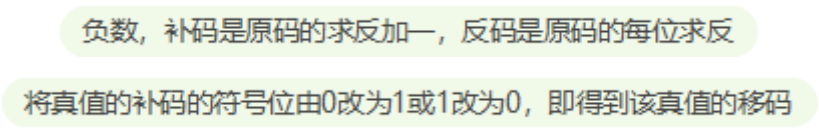
原码：1.000 0.000

补码：0.000

6.4假设有符号数的位数为8（包括符号位），补码能表示的真值的范围

补码能表示的真值范围为-128~+127

6.5 求反码以及移码的方法



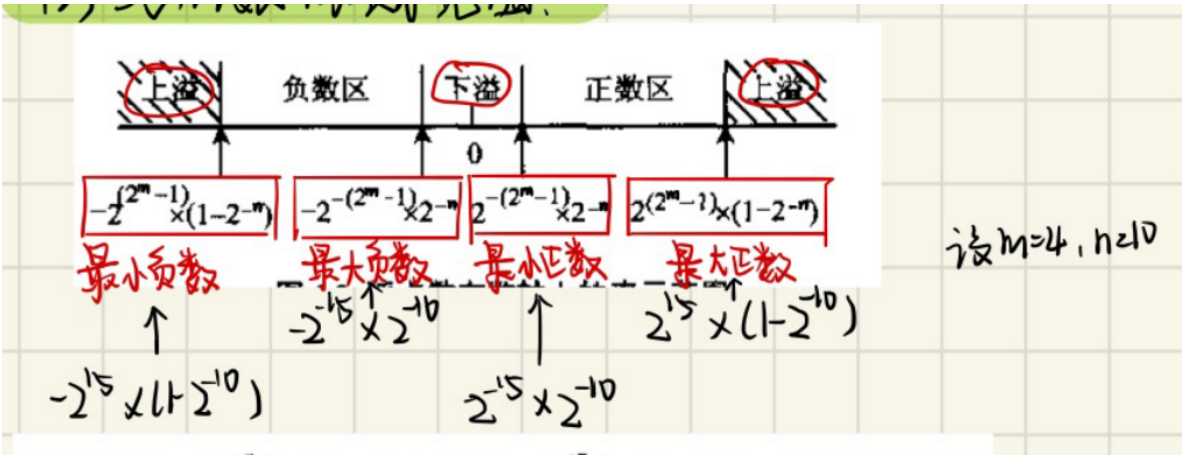
6.6 定点表示、浮点表示

1）**定点表示**：小数点固定在某一位置的数为定点数；
2）**浮点表示**：小数点位置可以浮动的数。

6.7 浮点数在机器中的表示形式，由哪几部分组成?

由尾数、数符、阶码、阶符四部分组成。

6.8 掌握规格化浮点数的表示范围（最大正数、最小正数、最大负数、最小负数）的计算方法



6.9 IEEE754标准规定的浮点数由哪几部分组成?

由数符、阶码（含阶符）以及尾数组成

按照 IEEE 754 标准，常用的浮点数的格式如图 2.14 所示。

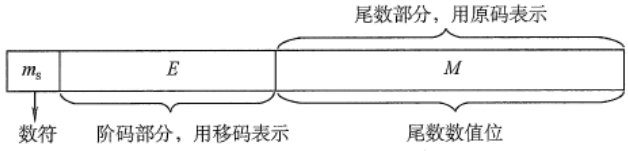


图 2.14　IEEE 754 标准浮点数的格式

IEEE 754 标准规定常用的浮点数格式有短浮点数（单精度、float 型）、长浮点数（双精度、double 型）、临时浮点数，见表 2.6。

表 2.6　IEEE 754 浮点数的格式

类型	数符	阶码	尾数数值	总位数	偏置值	
					十六进制	十进制
短浮点数	1	8	23	32	7FH	127
长浮点数	1	11	52	64	3FFH	1023
临时浮点数	1	15	64	80	3FFFH	16383

IEEE 754 标准的浮点数（除临时浮点数外），是尾数用采取隐藏位策略的原码表示，且阶码用移码表示的浮点数。

以短浮点数为例，最高位为数符位；其后是 8 位阶码，以 2 为底，用移码表示，阶码的偏置值为 $2^{8-1}-1=127$；其后 23 位是原码表示的尾数数值位。对于规格化的二进制浮点数，数值的最高位总是“1”，为了能使尾数多表示一位有效位，将这个“1”隐含，因此尾数数值实际上是 24 位。隐含的“1”是一位整数。在浮点格式中表示的 23 位尾数是纯小数。例如，$(12)_{10}=(1100)_2$，将它规格化后结果为 1.1×2^3，其中整数部分的“1”将不存储在 23 位尾数内。

注意：短浮点数与长浮点数都采用隐含尾数最高数位的方法，因此可多表示一位尾数。临时浮点数又称扩展精度浮点数，无隐含位。

6.10 IEEE754标准规定的浮点数中，阶码和尾数用什么形式表示?

阶码用移码表示，其偏移量是2^(n-1)，尾数用原码表示。

6.11 float占多少位？double占多少位

float为短实数，占32位，其中阶码8位，尾数23位。
double为长实数，占64位，其中阶码占11位，尾数为52位。

> **C语言中的浮点数类型及类型转换**

> C 语言中的float 和double 类型分别对应于IEEE 754 单精度浮点数和双精度浮点数。long double 类型对应于扩展双精度浮点数，但long double 的长度和格式随编译器和处理器类型的不同而有所不同。在C 程序中等式的赋值和判断中会出现强制类型转换，以char->int->long->double 和 float->double最为常见，从前到后范围和精度都从小到大，转换过程没有损失。
>
> 从int 转换为float 时，虽然不会发生溢出，但int 可以保留32 位， float 保留24 位，可能有数据舍入，若从int 转换为double 则不会出现。
>
> 从int 或float 转换为double 时，因为double 的有效位数更多，因此能保留精确值。
>
> 从double 转换为float 时，因为float 表示范围更小，因此可能发生溢出。此外，由于有效位数变少，因此可能被舍入。
>
> 从float 或double 转换为int 时，因为int 没有小数部分，所以数据可能会向0 方向被截断（仅保留整数部分），影响精度。另外，由于int 的表示范围更小，因此可能发生溢出。

6.12 对正数进行算术移位，当正数采用原码、补码、反码时，左移或右移时，低位或高位添补什么代码?

对于正数，其源码、补码、反码均等于真值，左移时，低位添补0，右移时，高位添补0。

6.13 对负数进行算术移位，当负数采用源码、补码、反码时，左移或右移时，低位或高位添补什么代码?

对于原码，左移或右移时，低位或高位均添补0；
对于补码：左移时，低位添补0，右移时高位添补1
对于反码：左移或右移时，低位或高位均添补1；

6.14 逻辑移位

逻辑移位是对**无符号数**的移位，由于无符号数不存在符号位，左移时，高位移丢，低位补零。右移时，低位移丢，高位补零。

6.15 加法和减法时，什么情况下可能发生溢出？如何简单判断发生溢出?

1）正数加正数，正数减负数，负数加负数，负数减正数时，可能会发生溢出。
2）如果参加操作的两个数符号相同（转换成补码的加法），其结果与源操作数符号不同，即为溢出。
3）如果补码采用1位符号位，如果最高有效位的进位和符号位的进位不同，则发生溢出。

6.16 定点乘法运算可以使用加法和移位来实现吗?

可以。

6.17 浮点加减运算基本按照哪几步来进行?

1）对阶：使小数点对齐；
2）尾数求和：将对阶后的两个尾数按照定点加减运算规则求和；
3）规格化：尾数规格化；
4）舍入：尾数右规时，丢失数值位；
5）溢出判断：判断结果是否溢出。

6.18 如何判断浮点运算结果是否溢出?

阶码是否超出了其表示范围。（使用2个符号位判溢出）

第七章指令系统

7.1 机器指令、指令系统

1）**机器指令**：每一条机器语言的语句；

2）**指令系统**：全部机器指令的集合

7.2 一条指令包含哪两个主要部分？请简要说明各部分作用

1）**操作码**：指明指令要完成的操作；
2）**地址码**：指明指令要操作的数据或数据来源；

7.3 操作码长度有固定长度和可变长度两种，各自有什么优点？

1）**固定长度**：便于硬件设计，指令译码时间短；
2）**可变长度**：压缩了操作码平均长度；

7.4 指令中地址码中的地址可以是哪些设备的地址？

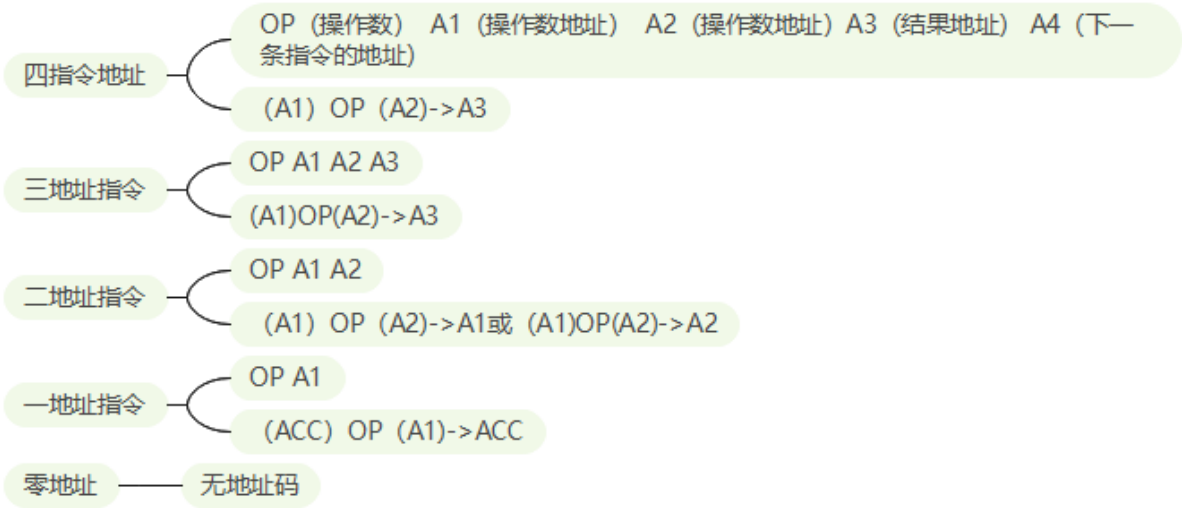
可以是**主存**地址、**寄存器**地址或**I/O设备**的地址；

7.5 指令中地址的个数可以有几个？

四地址、三地址、二地址、一地址以及零地址。

7.6 假设指令中有四个地址、三个地址、两个地址以及一个地址，各自需要访存几次？

1）四地址：访存4次；
2）三地址：访存4次；
3）两地址：访存3次；
4）一地址：访存2次；



7.7 当使用寄存器代替指令字中的地址码字段后，有哪些优点？

1）**扩大**指令字的寻址**范围**；
2）**缩短**指令**字长**；
3）**减少**访存**次数**

7.8 数据在存储器中存储时，为什么要按照边界对齐？

减少访存次数。

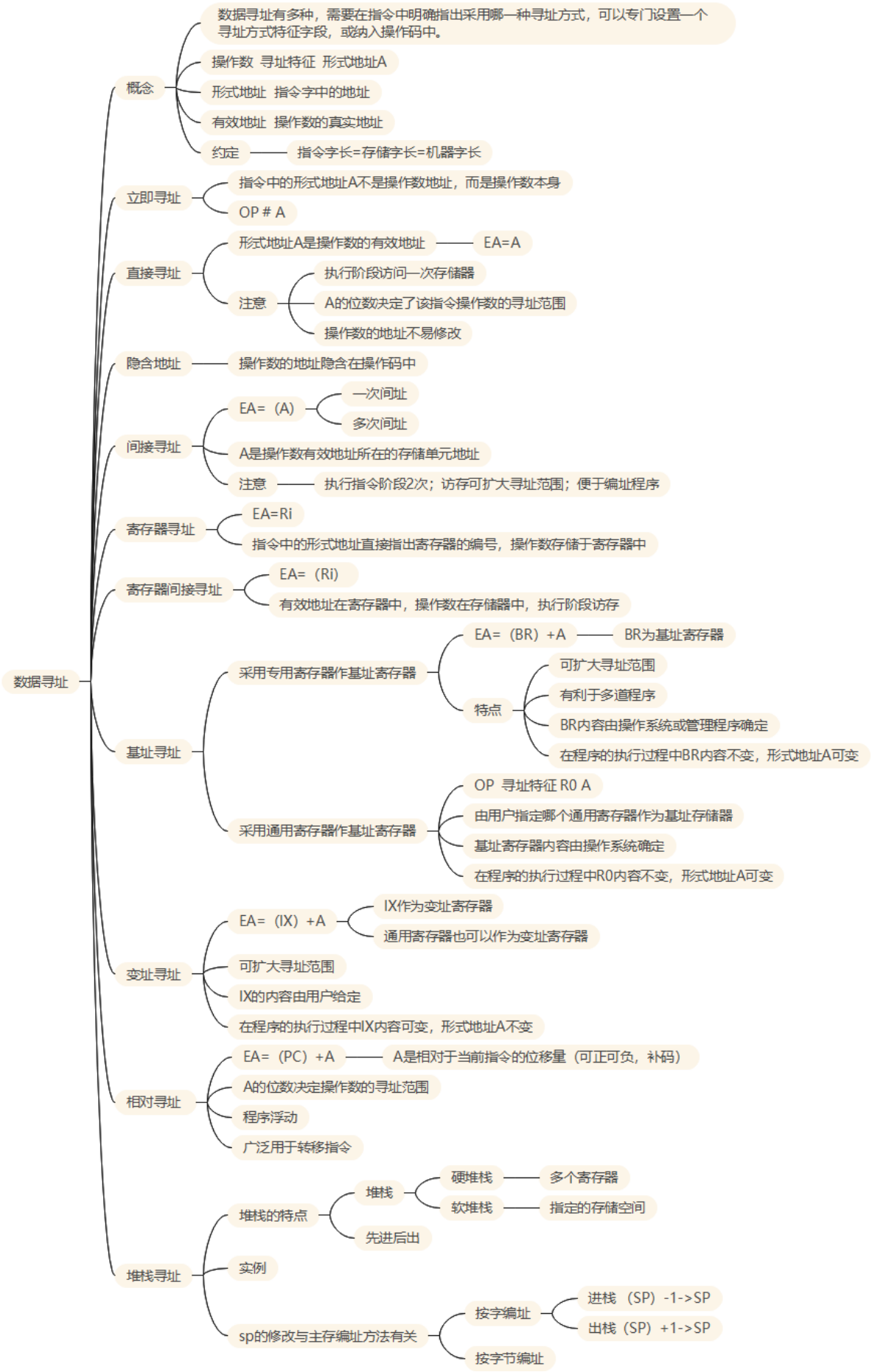
7.9 寻址方式包括哪两类？

1）**指令寻址**：下一条**将要执行的指令**的指令地址；
2）**数据寻址**：确定**本指令的操作数地址。**

7.10 什么是形式地址？什么是有效地址？

1）**形式地址**：指令的地址码字段通常都不代表操作数的真实地址，称为形式地址，记为A；
2）**有效地址**：操作数的真实地址，记为EA，由寻址特征和形式地址共同决定；

7.11 了解各种寻址方式的概念及根据形式地址形成有效地址的方式。



7.12 什么是RISC？什么是CISC？

RISC：精简指令集；
CISC：复杂指令集；

7.13 指令流水线的基本概念

流水线的基本原理

流水线技术是一种显著提高指令**执行速度与效率**的技术。

方法是:指令取指完成后,不等该指令执行完毕即可取下一条指令。

> 如果把一条指令的解释过程进一步细分,例如分成取指,译码, 访存，执行,访存和写回五个子过程,并用五个子部件分别处理这五个子过程.这样只需在上一指令的第一子过程处理完毕进入第二子过程处理时,在第一子部件中就开始对第二条指令的第一子过程进行处理.随着时间推移,这种重叠操作最后可达到五个子部件同时对五条指令的子过程进行操作.

五级流水CPU的各阶段

五级流水线指的是：**取指、译码、执行、访存、写回**五个操作。
1）**取指**：
指令取指（InstrucTIon Fetch）是指将指令从**存储器中读取**出来的过程。

2）**译码**：
指令译码（InstrucTIon Decode）是指将存储器中**取出的指令**进行**翻译**的过程。经过译码之后得到指令需要的**操作数寄存器索引**，可以使用此索引从通用寄存器组（Register File，Regfile）中将**操作数读出。**

3）**执行**：
指令执行是指对指令进行**真正运算**的过程。在"执行"阶段的最常见部件为**算术逻辑部件运算器**（ArithmeTIc Logical Unit，ALU），作为实施**具体运算**的**硬件功能单元。**

4）**访存**：
访存（Memory Access）是指**存储器访问指令**将**数据**从存储器中**读出，或者写入**存储器的过程。

5）**写回**：
写回（Write-Back）是指将**指令执行的结果**写回**通用寄存器组**的过程。如果是**普通运算指令**，该结果值来自于"执行"阶段计算的结果；如果是**存储器读指令**，该结果来自于"访存"阶段从存储器中读取出来的数据。

流水线方式的特点

1. 把**一个任务**（一条指令或一个操作）**分解**为几个有联系的**子任务**，每个子任务由一个专门的功能部件来执行，并依靠**多个功能部件并行工作**来缩短程序的执行时间。
2. 流水线每个**功能部件**后面都要有一个**缓冲寄存器**，或称**锁存器**，其作用是保存本流水段的**执行结果**，供给下一流水段使用。
3. 流水线中各功能段的**时间应尽量相等**，否则将引起堵塞、断流。
4. 在流水线中处理的必须是**连续任务。**
5. 流水线需要有**装入时间**和**排空时间。装入时间**是指**第一个任务**进入流水线到输出流水线的时间。**排空时间**是指**最后一个任务**进入流水线到输出流水线的时间。

影响流水线性能的因素

1）**结构相关**是当多条指令同一时刻**争用同一资源**形成冲突

> 解决方案：（1）暂停一个时钟周期（2）单独设置数据存储器和指令存储器

2）**数据相关**是指令在流水线中重叠执行时,当后继指令需要用到前面指令的执行结果时发生的冲突.

> 解决方案：（2）暂停一个时钟周期（2）数据旁路：把前一条指令的ALU计算结果直接输入到下一条指令

3）**控制相关**是当流水线遇到**分支指令**和其他**改变PC值**的指令时引起的.

解决方案：(1)**延迟转移技术。**将转移指令与其前面的与转移指令无关的一条或几条指令对换位置，让成功转移总是在紧跟的指令被执行之后发生，从而使预取的指令不作废。

(2)**转移预测技术。**

7.14 执行单条指令时单周期CPU和五级流水CPU谁更快？为什么？

1）五级流水CPU就是在多周期CPU的基础上+流水线思想-------->实现并行。
2）传统的单周期CPU是指指令在一个时钟周期（即时钟到来一次）内执行完成，包括所有的操作。但这里存在的确定就是时钟周期的时间是固定的，所以时钟周期的时间应该是**最长的指令所需要的时间**，对于那种很短的指令，就会造成浪费。引入**多周期**，这时就可以减少浪费的时间。
3）单周期CPU（是指在一个时钟周期内完成这五个阶段的处理）。因为五级流水CPU中每一个操作的流水时间是一样的，取得是这五步中的最大执行时间，那么就是最大操作时间*需要执行的操作数目（取指、译码、、、）。
4）**因为只有一条指令嘛，所以单周期最快，多条指令是考虑流水线比较好。**