# 啊我有兔子牙——体系结构复习

1. 第1章 体系结构的基本概念 10%
	1. 1现代计算机体系的基础，冯.诺依曼结构的特征（6）
		1. 运算器为中心
		2. 存储器按一维线性编址，顺序访问存储器的地址单元，每个存储单元的位数固定
		3. 程序存储 指令和数据无区别，存放在存储器中，指令和数据一样可以送到运算器中，进行运算，指令和数据的区别，主要在于地址区域不同
		4. 指令在存储器中按顺序存放，由一个顺序控制器来指定即将被执行的指令地址，每读取一条指令地址，计数器自动按顺序递增
		5. 指令由操作码和地址码组成，操作码指明操作类型，一直码指明操作数的地址和结果地址
		6. 数据以二进制表示
	2. 2把一条指令分解为五个部分



* 1. 3计算机体系结构 组成和实现
		1. 定义
			1. 计算机体系结构是程序员所看到的计算机属性，即概念性结构和功能特性
		2. 属性（9）
			1. 数据表示，硬件可以直接辨认和处理的数据类型
			2. 寻址规则 包括最小寻址单元 寻址方式及其表示
			3. 寄存器定义 包括各种寄存器的定义数量和使用方式
			4. 指令系统 包括机器指令的操作类型和格式指令间的排序和控制机构等
			5. 中断系统 中断的类型和中断响应硬件的功能等
			6. 机器工作状态的定义和切换，如管态和目态
			7. 存储系统 主存容量，程序员可用的最大存储容量的
			8. 信息保护 包括信息保护方式和硬件对信息保护的支持
			9. iO结构 包括io连接方式等
	2. 4计算机体系结构设计的三个方面
		1. 计算机硬件
		2. 计算机组成
		3. 计算机指令系统
	3. 5计算机系统中的层次



* 1. 6系列机和兼容
		1. 系列机是指具有相同的 体系结构，但具有不同的 组织和实现 的一系列不同型号的机器
		2. 根本特征 向后兼容
	2. 7设计和分析中最常用的几条基本原则和方法
		1. 大概率事件优先
		2. Amdahl定律
			1. 如果一个程序包括串行和并行，随着机器数量的增加，运行的速度会越来越快，趋近于0而，串行的速度不变
		3. 程序的局部性原理
			1. 计算机在程序执行过程中的一种规律，局部性分为时间局部性和空间局部性
			2. 时间局部性质近期被访问的代码很可能被再次访问
			3. 空间局部性是指地址上相邻的代码可能会被连续访问
		4. CPU的性能
	3. 简述计算机系统结构，计算机组成，计算机实现三者之间的关系
		1. 计算机组成 是计算机系统结构的 逻辑实现
		2. 计算机实现 是计算机组成的 物理实现
		3. 一种计算机系统结构可以有多种计算机组成，一种组成可以有多种计算机的实现
	4. 计算机系统的flynn（弗林）分类法分类
		1. 单指令流单数据流sisd （device） 指令部件只对一条指令处理，只控制一个操作部件操作，比如一般的串行单处理机
		2. 单指令多数据流simd 由单一指令部件同时控制多个重复设置的处理单元执行，同一指令下不同数据的操作 如阵列处理机
		3. 多指令流单数据流misd 单个指令部件对同一数据的各个处理阶段进行操作，这种机器很少见
		4. 多指令流多数据流，mimd 多个独立或相对独立的处理机，分别执行各自的程序作业或进程，例如多处理机
1. 第2章 指令结构 结构问题 14%
	1. CPU中用来存放操作数的存储单元
		1. 堆栈
		2. 累加器
		3. 通用寄存器
	2. 指令系统的结构可分为
		1. 堆栈型结构
		2. 累加器型结构
		3. 通用寄存器型结构
			1. 寄存器，寄存器型 rr
			2. 寄存器，存储器型 rm
			3. 存储器，存储器型 mm
	3. 成功的指令系统，应该能适应
		1. 硬件技术
		2. 软件技术
		3. 应用特性的变化
	4. 寻址方式
		1. 是指令系统中所产生的所要访问的 数据地址 的方法，对于 存储器操作数 来说，由寻址方式确定的存储器地址称为 有效地址
	5. 现代指令系统中，指令的功能由指令的操作码决定，从功能上看指令可分为
		1. 运算指令
		2. 访存指令
		3. 转移指令
		4. 特殊指令
	6. 通常有几种编码格式，它的适用范围是什么
		1. 变长编码格式
			1. 适用于关注目标代码的大小，而不是性能的情况下
		2. 固定长度编码格式
			1. 如果关注的是代码的性能，而不是目标代码的大小，就可以使用固定长度编码格式
		3. 混合型编码格式
			1. 需要同时兼顾目标代码的长度和降低译码复杂度时，就可以采用混合编码格式
	7. 计算机指令集的结构设计包含哪些内容
		1. 指令集功能设计
			1. 主要有r ISC和cisc两种技术发展方向
		2. 指令集格式的设计
			1. 变长编码格式
			2. 固定长度编码格式
			3. 混合型编码格式
		3. 寻址方式的表示
			1. 可以将寻址方式编码，在操作码中，也可以将寻址方位作为一个单独的部分来表示
		4. 寻址方式的设计
			1. 可以通过对基准程序进行测试统计，看各种寻址方式的使用频率，根据频率来设置对应的寻址方式
		5. 操作数表示和操作数类型
			1. 主要的操作数类型和操作数的表示可以选择浮点数据类型，整数数据类型，字符型等
	8. 写出CPU的性能公式，根据这个公式简单描述rISC指令及结构计算机和cisc指令集结构计算机的性能特点
		1. CPU的性能公式 CPU时间=IC\*CPI\*T
		2. IC为目标程序被执行的指令条数，CPI为指令平均执行周期数，T是时钟周期的时间
		3. 相同功能的CIS C目标程序的指令条数少于III的指令条数，但是C ISC的CPI和T都大于ri SC的CPI和T，因此ISC程序的执行时间比RISC的更长
2. 第3章 流水线 26%
	1. 什么是流水线
		1. 流水线技术是指将一个 重复的时序过程 分解成若干个 子过程，每一个子过程都可以有效的在它的专用功能段上和其他子过程【同时执行】的过程
	2. 流水线的特点（5）
		1. 流水过程有多个【相联系的子过程】组成，每个过程称为流水线的【级，段】，一条流水线的段数，也叫做流水线的【深度或流水深度】
		2. 每个子过程由【专门的功能段】实现
		3. 每个功能段的【所需时间尽量相等】
		4. 流水线需要有【通过时间】【排空时间】，只有经过通过时间后，流水线过程才能进入【稳定】的工作状态，流水线都不是满负荷工作
		5. 流水线技术适合【大量重复】的时序过程，只有在输入端能【连续】的提供任务时，它的效率才能【充分发挥】
	3. 流水线的分类
		1. 按照流水线的功能
			1. 单功能流水线
			2. 多功能流水线
		2. 按连接方式
			1. 静态流水线
			2. 动态流水线
		3. 流水的级别
			1. 部件级流水线
			2. 处理机级流水线
			3. 处理机间流水线
	4. 相关性
		1. 结构相关
			1. 指令在同步执行的过程中，硬件资源满足不了重叠执行的要求，发生了资源冲突
		2. 数据相关
			1. 一条指令需要用到前面指令的执行结果重叠执行的时候就会引起数据相关
		3. 控制相关
			1. 流水线遇到分支指令或者是其他能够改变PC值的指令的时候，就会发生控制相关
	5. 流水线解决瓶颈的方法有
		1. 细分瓶颈段
		2. 重复设置瓶颈段
	6. 减少流水线处理分支暂停时钟周期数的途径包括
		1. 尽早判断分支转移是否成功
		2. 尽早计算出分支成功转移的PC值
	7. 通过软件（编译器）来减少分支延迟有几种方法（4）
		1. 【冻结或排空】流水线的方法
			1. 一旦流水线的译码阶段【检测到】分支指令的话，就【暂停】后面的所有指令，直到分支指令【到达访存段】之后，确定他【成功并计算出新的PC】为止
		2. 预测分支失败的方法
			1. 我们【沿着】失败的分支【继续处理】指令，就好像什么都没有发生，当确定分支式失败时，说明预测正确，流水线正常流动
		3. 预测分支成功的方法
			1. ，当确定分支式成功时，流水线【按分支目标地址重新取指令执行】
		4. 延迟分支的方法
			1. 主要思想是从【逻辑上】来延长分支指令的执行时间，把延迟分支看成是由原来的若干个延迟草和分支指令构成，不管分支【是否成功】都要按【顺序执行】延迟槽中的指令
		5. 三种方法的共同特点是，他们对于分支的【处理方法】在程序执行过程中始终是【不变】的，他们总是要么预测分支成功，要么总是预测分支失败
	8. 解决流水线数据冲突的方法主要有哪4种
		1. 定向技术
			1. 在某条指令产生一个结果之前，其他指令并不需要去真正的计算结果，如果将【该结果从其产生的地】方【直接送到其他指令需】要他的地方就可以避免暂停
		2. 暂停技术
			1. 设置一个【流水线互锁的功能部件】，一旦流水线互锁【检测到数据相关】，流水线【暂停执行】，发生数据相关指令后续的所有指令，直到该数据相关【解决为止】
		3. 采用编译器调度
			1. 为了减少停顿，对于【无法用定向技术解决】的数据冲突，可以通过在【编译时】让编译器实现【指令调度或流水线调度】来消除冲突
		4. 重新组织代码顺序
			1. 重新组织指令顺序，以加大【相关指令的距离】，进而【减少】数据冲突的【可能性】
3. 第4章 指令级并行 10%
	1. 程序中的相关主要分为
		1. 名相关
			1. 指令使用的寄存器或存储器称为名，如果有两条指令使用相同的名，但是它们之间没有数据流，就叫做名相关
				1. 反相关

先读后写相关

* + - * 1. 输出相关

先写后写相关

* + 1. 数据相关
		2. 控制相关
	1. 开发并行性是为了进行【并行处理】并行性又包括
		1. 同时性
		2. 并发性 二重含义
	2. 编译器在完成指令调度时，一般受限于两个特性，分别是程序固有的
		1. 指令级并行性
		2. 流水线 功能部件 的 执行延迟
	3. 为了允许乱序执行指令级并行，可以将基本流水线的译码阶段再细分为
		1. 流出
			1. 检查是否存在结构阻塞
		2. 读操作数
1. 动态调度算法 指令及并行里的多条指令流出技术
	1. 记分牌技术
		1. 所有的指令在流出阶段是顺序的，但是在第2阶段读操作数时，只要指令运行所需要的资源满足，并且没有数据阻塞，就应该允许指令乱序执行
	2. Tomasulo算法
		1. 消除了数据的写后，写后写，和先读后写相关导致的拥塞
		2. 主要思想
			1. 只要【操作数有效】，就把它取到我们的【保留站】，避免指令流出的时候，才到计算器里面去取数据，这就是【即将执行的指令】，从相应的【保留站中取得操作数】，而不是从寄存器中
			2. 指令的【执行结果】也是直接送到等待数据的【其他保留站】，因而，对于连续的寄存器写，只有【最后一个】才真正的【更新到寄存器】中的内容。
			3. 一条【指令流出】时【存放操作数的寄存器名】被替换成对应于该寄存器【保留站的名称】 或者是编号
			4. 【指令流出逻辑】和【保留站】相结合，实现【寄存器换名】从而完全消除了寄存器数据，写后写和先读后写相关的这类名相关
		3. 名词解释 指令级并行指令调度指令的动态调度
			1. 指令级并行
				1. 简称【il P】是指【指令之间】存在的一种【并行性】，计算机利用它可以并行执行【两条或两条】以上的指令
			2. 指令调度
				1. 通过在【编译】时让【编译器】重新组织【指令顺序】，或通过【硬件】在执行时【调整指令来消除】冲突
			3. 指令的动态调度
				1. 指在保持【数据流和异常行为】的情况下，通过【硬件】对【指令顺序】进行重新安排以提高流水线的利用率，减少停顿现象。它由硬件在程序实际运行时实施
2. 第5章 存储层次 18%
	1. 存储器的三个主要指标是
		1. 容量速度和每位价格
		2. 当前计算机系统的存储系统是一个层次，结构各层次分别为
			1. 通用寄存器
			2. 高速缓存
			3. 主存
			4. 辅存
			5. 脱机大容量存储器
			6. 存储层次的4个问题
				1. 映像规则

把一个块调入高一层的存储器，可以放在哪些位置上

全相连方式

组成块可以被放到任意一个K10块的位置

直接相连方式

储存块只能被放置到唯一的一个cash块的位置

组相联方式

可以放在唯一的组中任何位置

* + - * 1. 查找算法

当要访问的快再高，一层存储器是如何找到该快

随机法

先进先出，

最近最少使用，

最不常使用法

* + - * 1. 替换算法

发生失效时应该替换哪一块

* + - * 1. 写策略

进行写访问时进行哪些操作

* + 1. 两种层次关系中 cache和储主存是为了弥补主存速度的不足储存，主要由硬件实现
		2. 辅存和主存是为了弥补主存容量的不足，主要由软件实现
1. 第6章 输入输出
	1. 计算机io系统的吞吐量又被称为io带宽
	2. 响应时间又被称为响应延迟
	3. 计算机系统的基本输入输出方式包含程序查询，输入输出方式，中断输入的输出方式和DMA方式