前言：计算机组成原理课程内容比较多，但是考试内容只占一部分。因为本材料主要目标是攻克考试，所以接下来的内容基本以对老师复习指南（其实上面有很多例题，都做一遍就稳了）的进一步阐释和总结为主，而指南之外的内容仅做适当拓展。知识点之后我加入了自己的一些预测，仅是个人观点，做个参考即可，出现问题，别来找我。

目录

[一、计算机系统概述 2](#_Toc13339529)

[二、数据的机器级表示 2](#_Toc13339530)

[1.浮点数尾数的规格化 2](#_Toc13339531)

[2.数据校验码：奇偶校验码、海明校验码、循环冗余校验码 3](#_Toc13339532)

[三、运算方法和运算部件 3](#_Toc13339533)

[1.ALU 4](#_Toc13339534)

[2.定点数一位乘法 4](#_Toc13339535)

[3.浮点数的加减法运算 4](#_Toc13339536)

[四、指令系统 5](#_Toc13339537)

[1. 指令格式设计 5](#_Toc13339538)

[2. 指令系统设计——寻址方式 5](#_Toc13339539)

[3. 指令系统设计——操作码编码 7](#_Toc13339540)

[4. 掌握MIPS汇编 7](#_Toc13339541)

[五、中央处理器 8](#_Toc13339542)

[1.指令周期、数据通路等概念 8](#_Toc13339543)

[2.单周期、多周期处理器 8](#_Toc13339544)

[3.硬连线路控制器和微程序控制器的设计 8](#_Toc13339545)

[4.异常和中断 8](#_Toc13339546)

[六、指令流水线 9](#_Toc13339547)

[1.五段流水线： 9](#_Toc13339548)

[2.流水线处理器的实现： 9](#_Toc13339549)

[3.冲突/冒险： 9](#_Toc13339550)

[七、存储器分层结构 13](#_Toc13339551)

[八、互联以及输入输出组织： 13](#_Toc13339552)

# 一、计算机系统概述

第一部分知识点简单，主要包括计算机层次结构的一些概述，硬件与软件、系统软件与应用软件、五大部件等，阅读书上5至8页，理解、简单记住即可。

关于CPU如何区分从存储器中取出的是指令还是数据：取指令周期取出的是指令，取数据周期取出的数据。

计算机语言分类：高级语言（大部分的编程语言）、低级语言（机器语言、汇编语言）

ISA：指令集体系结构，软件硬件之间接口的一个完整定义，是软件能够感知的部分，也称软件可见部分。

ISA是对指令系统的一种规定或结构规范，具体实现由微体系结构实现。

微体系结构（微架构）：微体系结构是软件不可感知的部分。相同的ISA可以拥有不同的微体系结构。

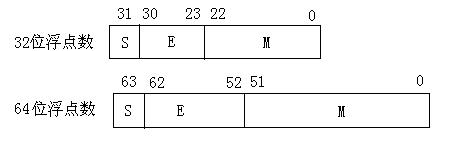
第一章大部分内容理解简单记住即可，以上内容建议通读课本P5-P12。

# 二、数据的机器级表示

本章内容看似复杂，实则考试以应用为主，难度不大，强烈建议下笔演练几次，熟能生巧。另外建议再熟悉下其余的数字表示方法：源码表示、补码表示、移码表示等，因为之后的第三章会用到。

## 1.浮点数尾数的规格化

浮点数的表示可以表示拆分成三部分，符号位（S）、数值为（M）、指数位（R），为了方便各个计算机之间通用一套表示方法，有了IEEE754标准，即浮点数的规格化。



对于32位浮点数，要求：符号位1位，阶码8位用移码表示，尾数23位用原码表示，其中尾数部分第一位总为1，所以被省略称为隐藏位。这样23位有效数字尾数表示了24为有效数字。双精度的用52位表示了53位。IEEE754中移码的偏置常数为2n-1-1，即127和1023。具体规格化流程之后的第三章会有体现。

对特殊数的解释：见书P40-P44。

*并不准的预测：浮点数的规格化问题很有可能会和之后的运算方法一章结合，进行浮点数加减法的运算等，出一道大题，所以强烈建议熟练浮点数的规格化！*

## 2.数据校验码：奇偶校验码、海明校验码、循环冗余校验码

掌握奇偶校验码、海明校验码的编码求法，熟悉三个校验码的区别、发现错误、纠错能力等。

校验原理：以同样的算法对使用前后的数据添加冗余校验位，进行比较，详见P54-P55。

奇偶校验码：只能发现奇数位错误，不能判定发生错误的位置，不具有纠错能力。

奇校验码：数据中有奇数个1是校验位P为0；偶数个1时P为1。

偶校验码：数据中有偶数个1是校验位P为0；奇数个1时P为1。

海明校验码：能发现一位错，并具有纠错能力

掌握求法，以及根据给出的校验码判断出错情况。P56-P58

循环冗余校验码：详见书上P59页，理解即可。

参考习题：16、19

*并不准的预测：海明校验码的求法很有可能会出现一道大题，一定要掌握。奇偶校验码要准确记住奇偶的区别，循环冗余检验码理解即可。*

# 三、运算方法和运算部件

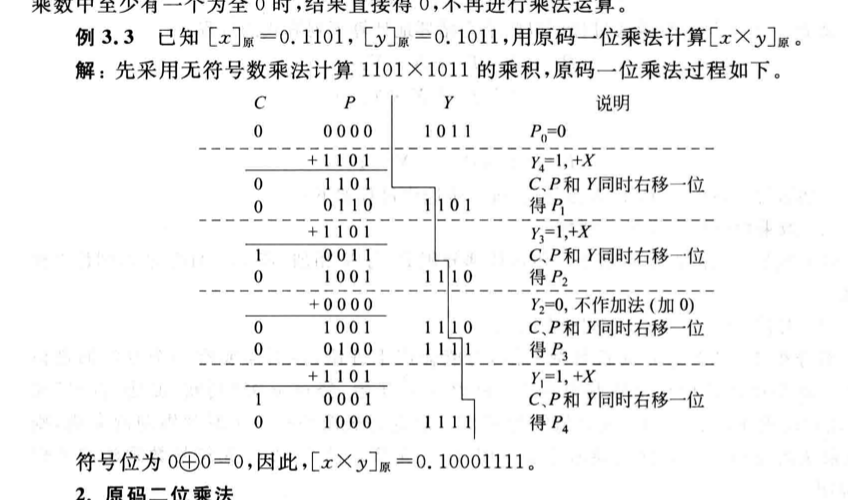
本章内容重点是原码一位乘法和补码一位乘法的计算，以及浮点数的加减运算。与上一章相似，看起来很难，理解之后，多实际运算几次，做做例题，熟能生巧，问题不大。关于上学期学过的原码、补码加减运算、加法器的构成等知识再看一看理解一下。

## 1.ALU

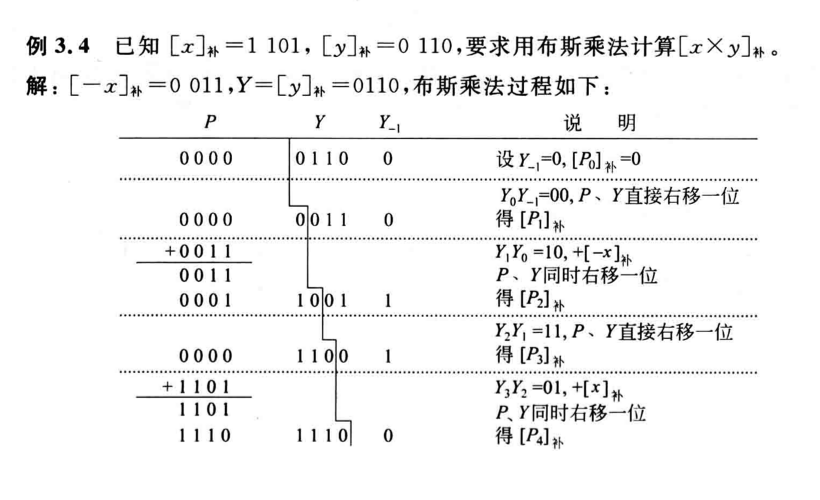
完成基本逻辑运算和定点数加减运算，核心部件是加法器。P70

## 2.定点数一位乘法

原码乘法运算：原码一位乘法



补码乘法运算：补码一位乘法



## 3.浮点数的加减法运算

流程：对阶->尾数加减->尾数规格化->尾数的舍入处理->溢出判断

注意点：

对阶时向大数对阶，注意对阶之后小数点的偏移；

尾数加减时注意吧隐藏位还原到尾数部分；

加减之后进行规格化，进行右规或者左规直到将1移到小数点左边。

尾数的舍入，增加两个附加位，按照一定规则舍入，保证精度。

溢出判断：主要判断阶码即指数部分即可。观察在左规或者右规期间是否发生了阶码上溢或者下溢。

例题：P95例三

参考习题：3、12

*并不准的预测：原码一位乘或者补码一位乘很有可能出现一个大题，这里很容易眼高手低，所以依然建议做一做例题和课后习题，浮点数加减也是如此，总之，计算这部分多做题！*

# 四、指令系统

本章要求熟悉MIPS汇编语言，注意与之前学过的AT&T的区别和相似之处，能用MIPS汇编语言表示C语言程序。掌握不同的寻址方式；区分不同的指令格式并会计算指令条数

## 指令格式设计

本节主要介绍了指令需要包含的信息和指令设计的基本要求与原则。注意理解三地址、二地址、一地址等概念和表示。P107-P108

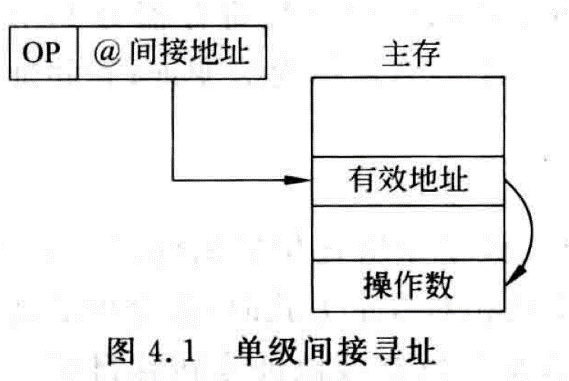
## 指令系统设计——寻址方式

本节的重点在于几个寻址方式：立即寻址、直接寻址、间接寻址、寄存器寻址、寄存器间接寻址、变址寻址、相对寻址、基址寻址以及其他寻址方式。

还要注意两个概念：形式地址与实际地址：形式地址指的是指令中给出的地址码字段，他不一定是真的地址，可能是基准地址还可能是偏移量。

立即寻址、直接寻址、寄存器寻址都比较简单，都是在指令中直接给出立即数或者地址等信息。

间接寻址和寄存器间接寻址相似，指令给出的地址码是存放有操作数有效地址的地址，即要寻找两次才可以找到真正的操作数。（寄存器间接寻址的形式地址是寄存器的编号。）

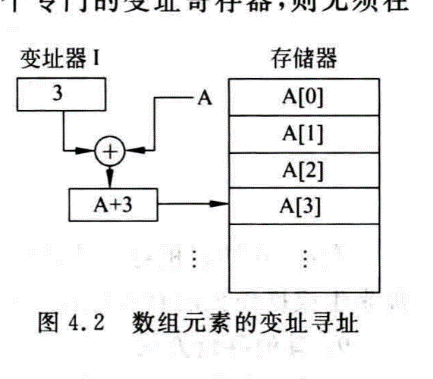


难点在于变址寻址、相对寻址和基址寻址。

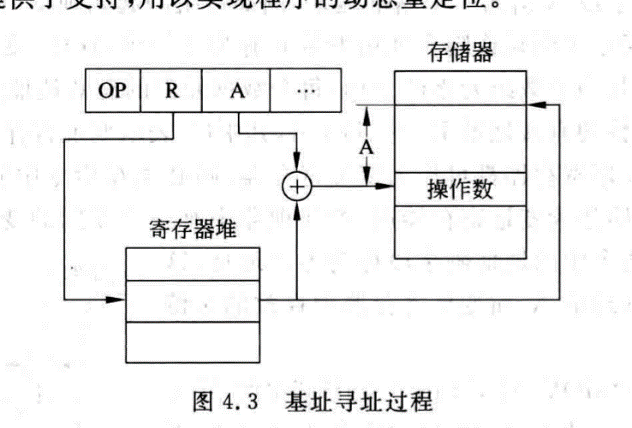
变址寻址：常用于数组元素的访问，给出的形式地址是基准地址A，偏移量存在变址寄存器中，实际地址的计算为 实际地址=基准地址+偏移量

例如，“MOV AL，【SI+1000H】，SI为变址寄存器，实际地址就是SI的内容加上1000H内容”，然后SI的内容会根据数组元素的长度自动加减。

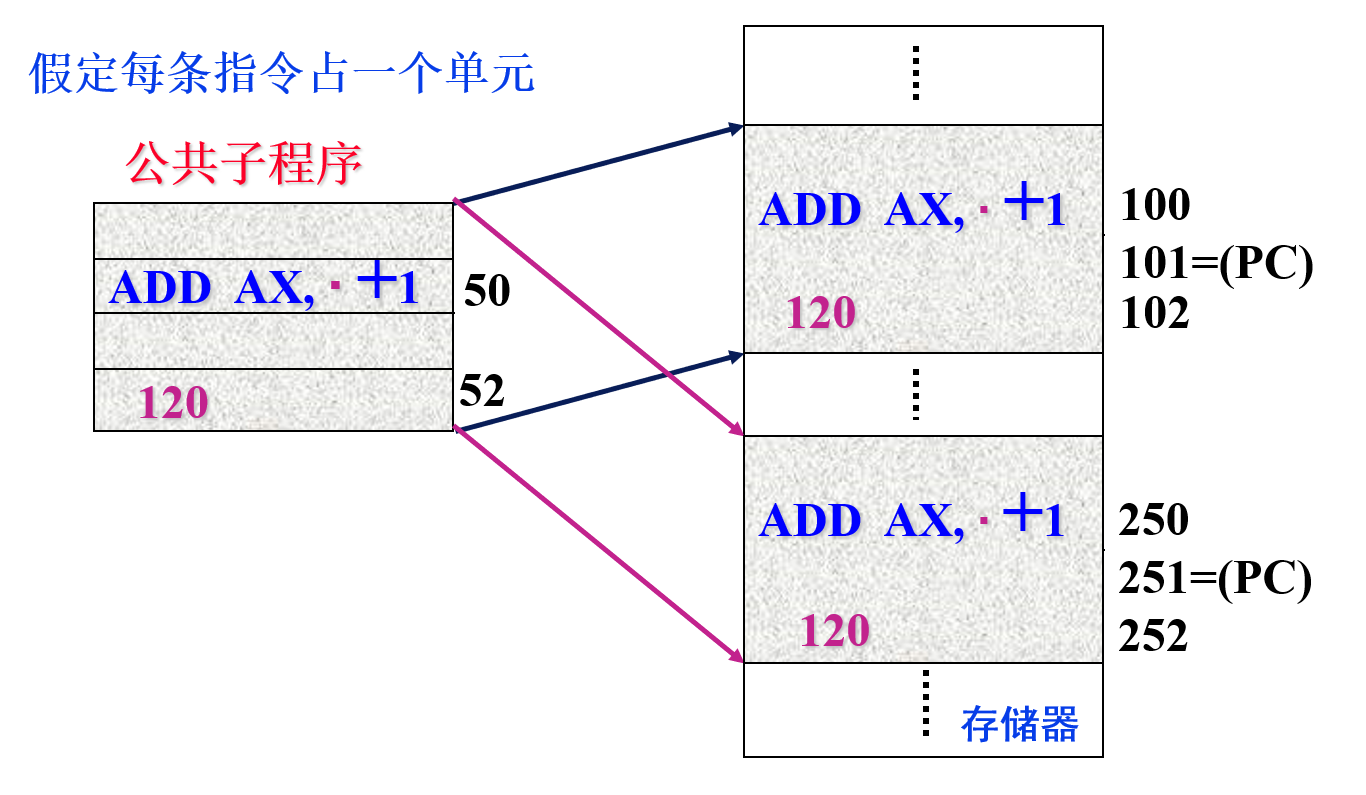
以此图为例子：假设每个元素占一个字节，执行语句for（i=0;i<n;i++），最初I等于0，取出A[0]之后，I自动加一变为1，然后取出A[1]，以此类推。



与它相对的概念是基址寻址。基址地址中给出的形式地址是偏移量，而基准地址明显或隐含地由基址寄存器B给出，实际地址 = B中地址 + 偏移量。这种寻址方式主要可以用于程序的重新定位，不需要每次都输入新的基准地址。

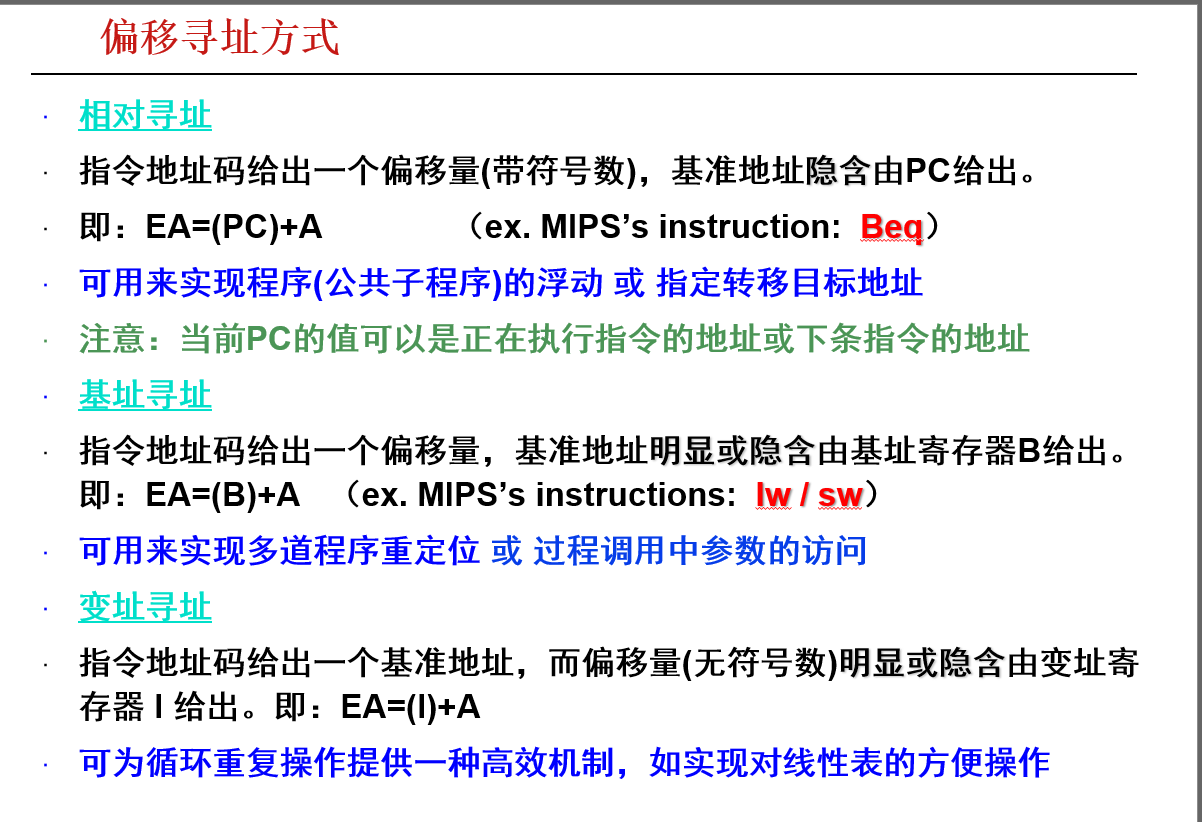


最后是相对寻址的概念。相对寻址的使用场景是某指令的操作数或转移目标地址与该条的指令的相对位置不变，总是位于该指令的前后固定位置上，这种情况下我们给出偏移量A作为形式地址，（注意A是一个带符号数），基准地址由PC隐含地给出。适合用于公共子程序的浮动或者相对转移。



关于几个寻址方式，尤其后面几个要注意他们的形式地址是什么，在哪里给出，实际地址如何计算，以及他们的应用场所，看的时候建议思考应用场景或者寻找合适的例子来理解。

而且我发现课程PPT这段讲得特别清楚，下图是PPT上对于这三种寻址方式的总结和对比。（第四章PPT第16页）



## 指令系统设计——操作码编码

这里设计的另一个重点习题是操作码的编码：定长操作码编码和拓展操作码编码，其中拓展操作码编码是考点。这样题的基本思路就是按照二地址->一地址->零地址的进行编码，认真理解书上的例4.1和课后6题，并不难。

## 掌握MIPS汇编

这个要注意MIPS汇编语言的指令的格式，比如指令的操作数在指令的后面。清楚什么是R、I、J型指令；熟悉常见的MIPS汇编指令（P127表格），能根据C语言指令写出其汇编指令、根据汇编反汇编出C语言指令。（至于由汇编得到指令机器代码指南上没有要求，但也可以看看）具体内容大家还是看书和做题吧。

最后两个指南上的杂项：（老师特意加上的，懂我意思吧）

CISC和RISC的特点 P119

计算机执行程序所需要的时间P= I×CPI×T 其中I是指令数,CPI是执行每条指令所需的平均周期数,T是时钟周期。

*并不准的预测：应该会有关于转移地址和MIPS汇编的题，还有确定指令条数，很有可能会考。*

# 五、中央处理器

这一章也属于比较重要的一章了，尤其微程序设计这里是必考的。

1.指令周期、数据通路等概念 P142-P143

大家看看理解就可以了。

程序计数器(PC)和指令寄存器(IR)等寄存器的功能与作用。

2.单周期、多周期处理器

单、多周期处理器的CPU设计、特点和比较。

书上内容好多，我个人认为，涉及到具体设计，就是与电子电路这方面知识相关的，可以适当略过，重点在于他们的原理，对比等，还有多周期状态的转移表，单周期时钟周期的确定等知识。

3.硬连线路控制器和微程序控制器的设计

这里其实很重要的，但是在这真的不好说清楚，大家多看书PPT多做题吧。下面是考点和例题。

根据CPU数据通路图，能够机器指令与微指令的关系。根据具体CPU数据通路图，能够给出具体指令周期流程图并给出微操作控制信号序列。P182-P184

采用断定法(下址字段法)确定下条微指令的地址，则微指令中下址字段的位数

机器指令与微指令的关系。根据具体CPU数据通路图，能够给出具体指令周期流程图并给出微操作控制信号序列。

微程序控制器与硬布线控制器的区别(执行速度，指令修改上)。

## 4.异常和中断

P184-P189 这要主要还是以理解为主：什么是异常、什么是中断。能根据例子看出是中断还是异常。

内部异常：指处理器内部异常引起的意外事件，分为硬故障中断和程序性异常，硬故障中断是指楹联线路出现异常，如内存掉电、存储器线路错误等；程序性异常又称软中断，由于CPU执行某个指令发生的处理器内部的异常事件。如整除0，断点、访问超时、非法操作码等。

内部异常分为：故障、自馅、终止三类

外部中断：外设完成任务或发生某些特殊事件，会向CPU发生中断请求。这些请求与指令无关，由CPU外部的I/O设备发出，通过外部中断线向CPU请求。中断是计算机外设的一种输入输出方式。

书上规定CPU内部异常事件叫做异常、外部中断事件叫做中断。

*并不准的预测：重点还是在断定法，很可能会有一个大题，也不太直观，重点复习一下。*

# 六、指令流水线

本章主要讲解流水线这种指令处理方式，这种方式可以用来提到执行效率，掌握五段流水线的含义、执行时间的计算以及三种冒险，尤其是Load-use数据冒险，课后4、6、7。

## 1.五段流水线：

IF, ID, OF, EX, WB以及N条指令执行的总时间——P194

以Load指令为准，分为五个阶段:

取指令段(IF)

取指令、计算PC+4

译码/读寄存器(ID/RF)段

指令译码、读Rs和Rt

执行(EXE)段

计算转移目标地址、ALU运算

存储器(MEM)段

读或写存储单元

写寄存器(Wr)段

ALU结果或从DM读出数据写到寄存器

均衡时指令吞吐率提高N倍，但不能缩短一条指令的执行时间

流水段数以最复杂指令所需步骤数为准（有些指令的某些阶段为空操作），每个阶段的宽度以最复杂阶段所需时间为准（尽量调整使各阶段均衡）

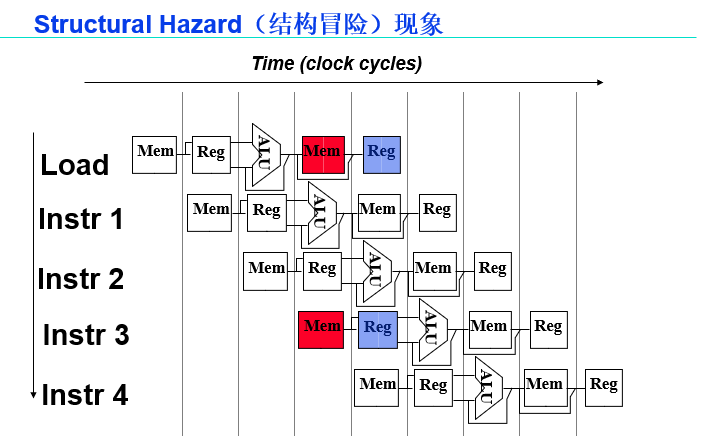
## 2.流水线处理器的实现：

复习指南没有。。。。。

## 3.冲突/冒险：

结构冒险、数据冒险、控制冒险，其中最重要的是数据冒险。

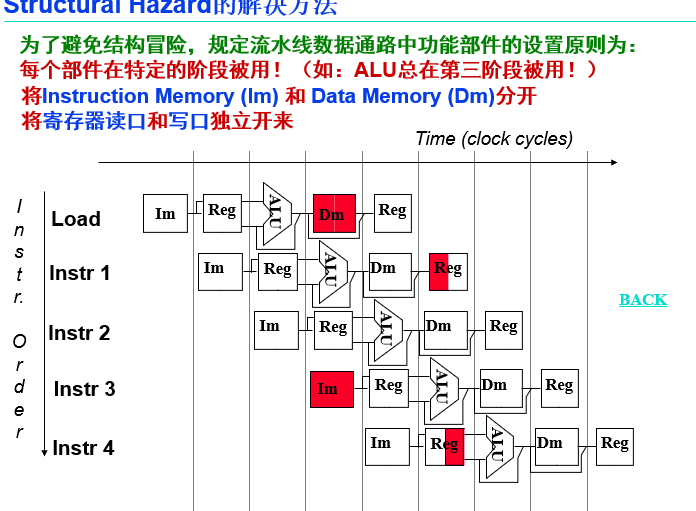
1.结构冒险：指的是硬件资源上的冲突，引起这一冒险的原因主要是同一部件同时被不同指令所使用，



处理方式：

1.功能划分原则，一个部件每条指令只能使用一次，且只能在特定时钟周期使用

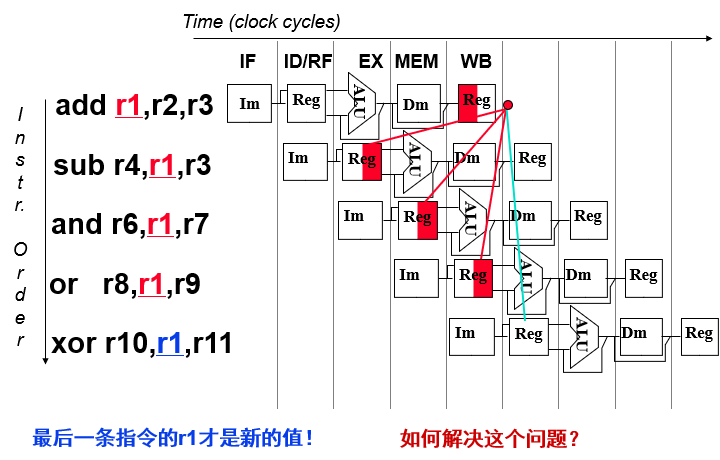
2.通过设置多个独立的部件避免硬件资源冲突，例如对于寄存器冲突访问，可将寄存器读口和写口独立开来，对于存储器访存冲突，将指令存储器和数据存储器独立开来，从而使指令和数据的独立访问。



事实上，现代计算机都引入了cache机制而且LI cache通常采用数据cache和代码cache

分立的方式，避免结构冒险。

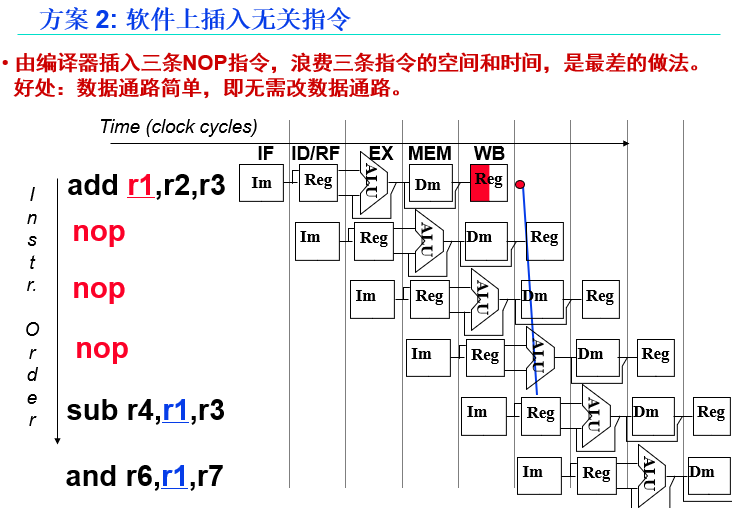
2.数据冒险：引起原因是后面指令用到了前面指令中还没有产生的数据。数据冒险分为多种，我们主要研究RAW，即写后读冒险。如下图所示，从第二道第四条指令所用的r1值在第一条指令结束之后才会产生，所以他们的r1都是旧值，只有最后一条的r1才是新值。



有以下几个措施：

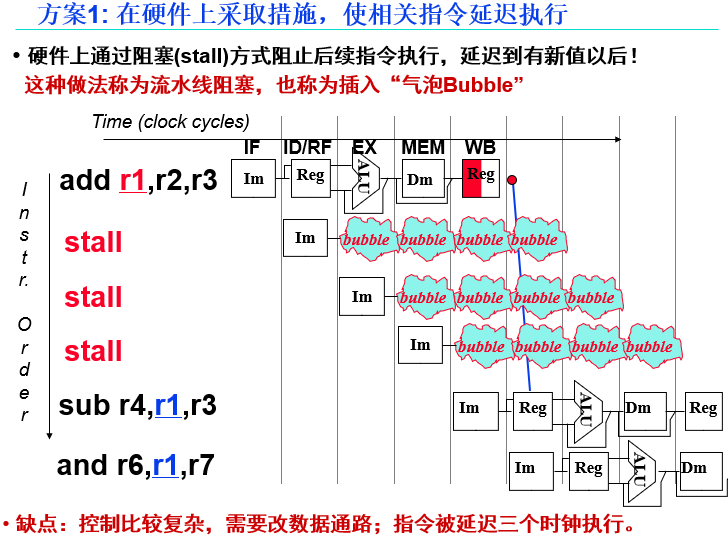
1. 插入空操作指令；

在软件上采取操作，在编译时预先插入空操作指令nop，这样做的好处是硬件控制简单，但浪费了指令存储空间和指令执行时间。



1. 插入气泡；

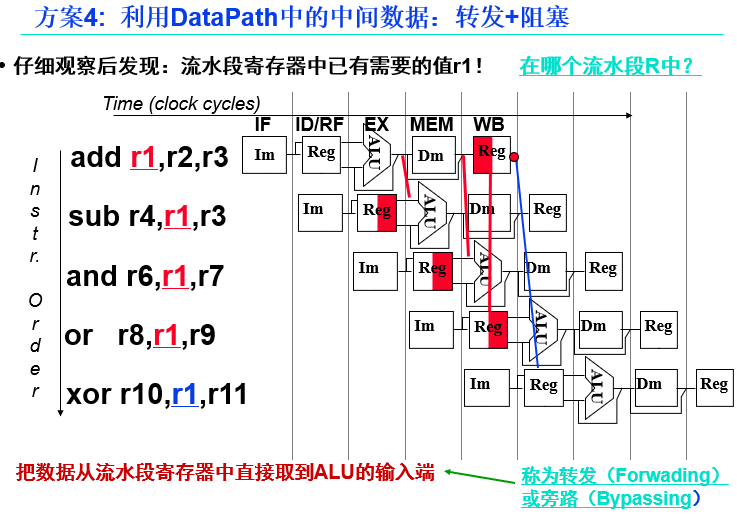
在硬件上采取措施，采用硬件阻塞（stall）的方式组织后续指令执行，这种方式被称为“插入气泡(bubble)”



1. 同一周期内寄存器堆先写后读： （来自老师PPT）

寄存器的写口和读口分别在前后半个周期进行操作，使写入数据被直接读入，只能解决部分冒险！

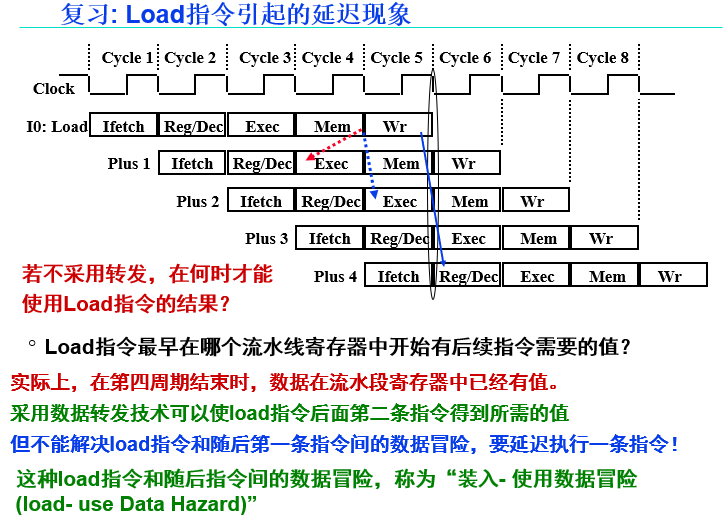
1. 采用转发技术：



我们可以发现实际上r1的值在第一条指令的EX阶段就已经有了，琐碎我们可以在这时将它取出放到ALU‘的输入端，这种技术被称为转发或者旁路技术。

转发技术必须在硬件上进行改动。

转发可以解决大部分的冒险，但是无法解决load之后立即需要值这种冒险。下面这个情况下，直到load的第四阶段才有值无法转发到下一指令的ex之前，这就是load-use冒险。



1. Load-use冒险的检测和处理：

最好的解决办法就是在编译时进行优化，通过调整指令顺序load-use现象。（例题6.1）

Load-use冒险发生的条件时：上一条是load指令，并且在存储器装入寄存器的数据是当前指令的源操作数。

3.控制冒险：略

*并不准的预测：可能会有流水线的计算题，计算时长；冒险这里要仔细读一读课本理解是怎么操作的，很可能会考察通过改变指令顺序处理load-use指令，做一下课后4、6、7。*

# 七、存储器分层结构

这一章内容多而杂，这是老师发的考点，大家结合这个复习就可以了。

1.存储器的分类。ROM和RAM叙述理解。存储器的速度从快到慢：寄存器—cache—主存—辅存。

2.SRAM和DRAM的区别(刷新：以行为单位)。P240

3.存储器芯片的扩展：位扩展、字扩展、字位扩展，需要的片数。P247

4.多模块存储器(或叫多体交叉存储器)、双口存储器(或叫双端口存储器)设立的目的

4.局部性：时间局部性和空间局部性。

5.Cache的行，槽 就是块。通过硬件来实现映射。 有效位，标记（三种映射变化）。

6.Cache命中率计算公式，平均访问时间公式。

7.直接映射、全相联、组相联映射方式的区别及使用案例(结合例题和习题)。

要求掌握给出主存块号或者主存单元地址，给出相应方式下(考核直接映射、组相联映射)的Cache块号。

8.Cache总容量的计算：有效位、标记、数据块。

9.Cache中主存块的替换算法：先进先出、LRU算法、最不经常用算法、随机替换算法。结合习题、例题。

10.虚拟存储器的TLB和Cache的访问过程。

段式、页式、段页式。

虚拟存储器管理机制中地址转换描述。

**习题参考部分**：第4、10、17题。

# 八、互联以及输入输出组织：

1.常见的输入输出设备，外部存储设备。

2.I/O接口的含义，I/O端口的含义。

3.I/O数据传送控制方式：程序直接控制方式，程序中断方式，DMA方式。其中DMA主要适合于磁盘等高速设备的数据传送。

最后悄悄地祝看到这的同学机组90+。