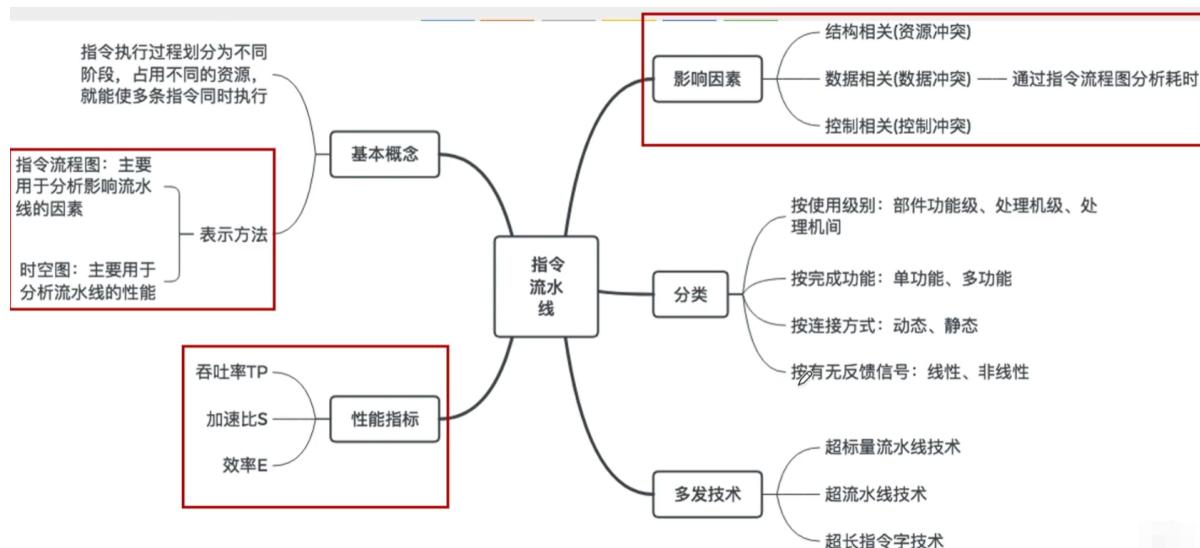
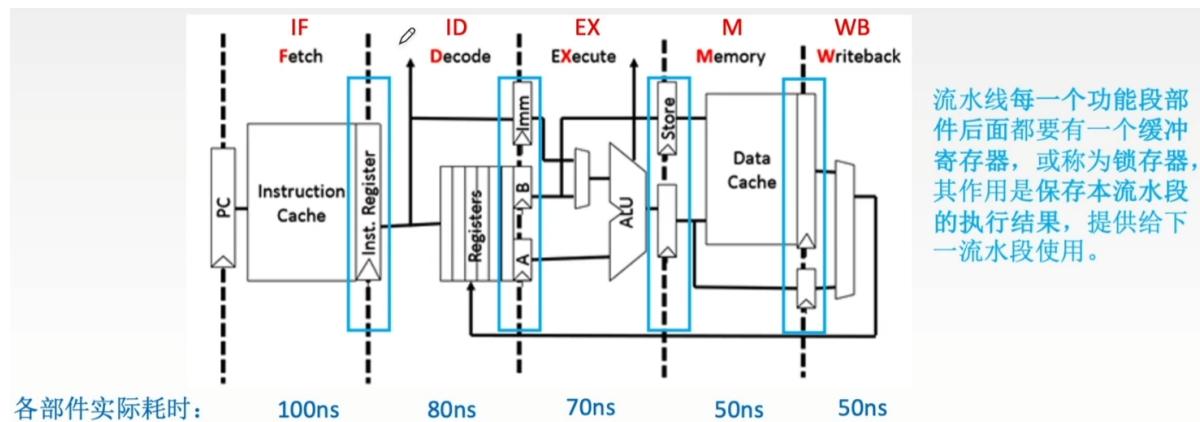


流水线的基本实现

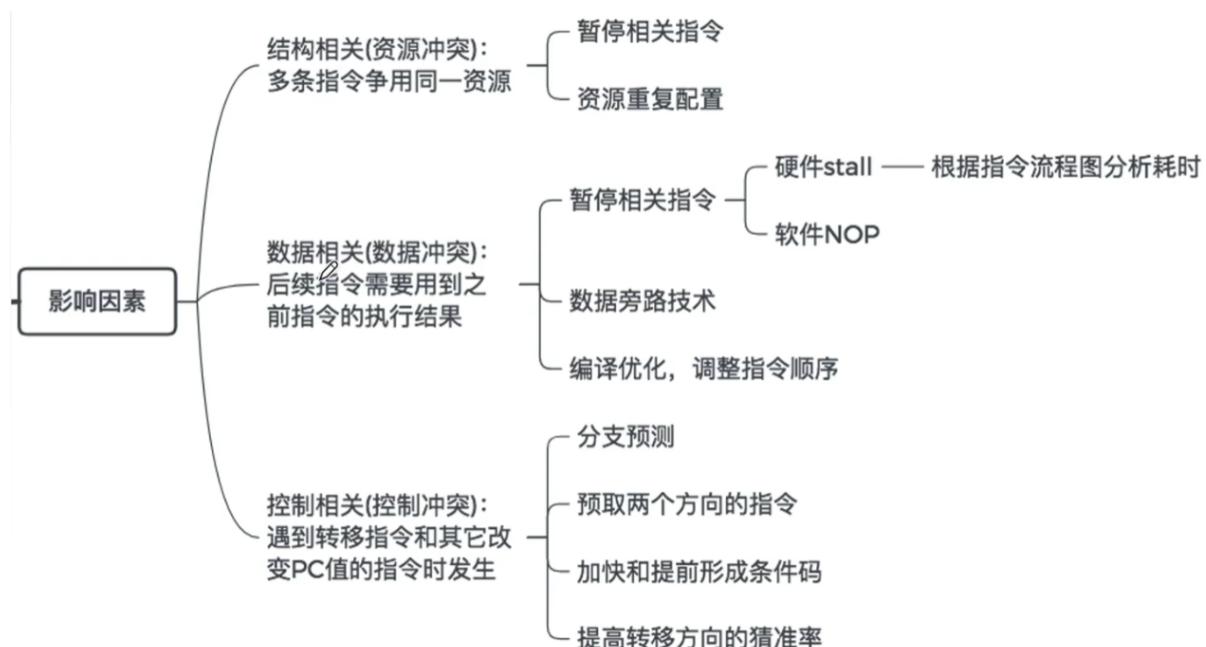


机器周期的设置



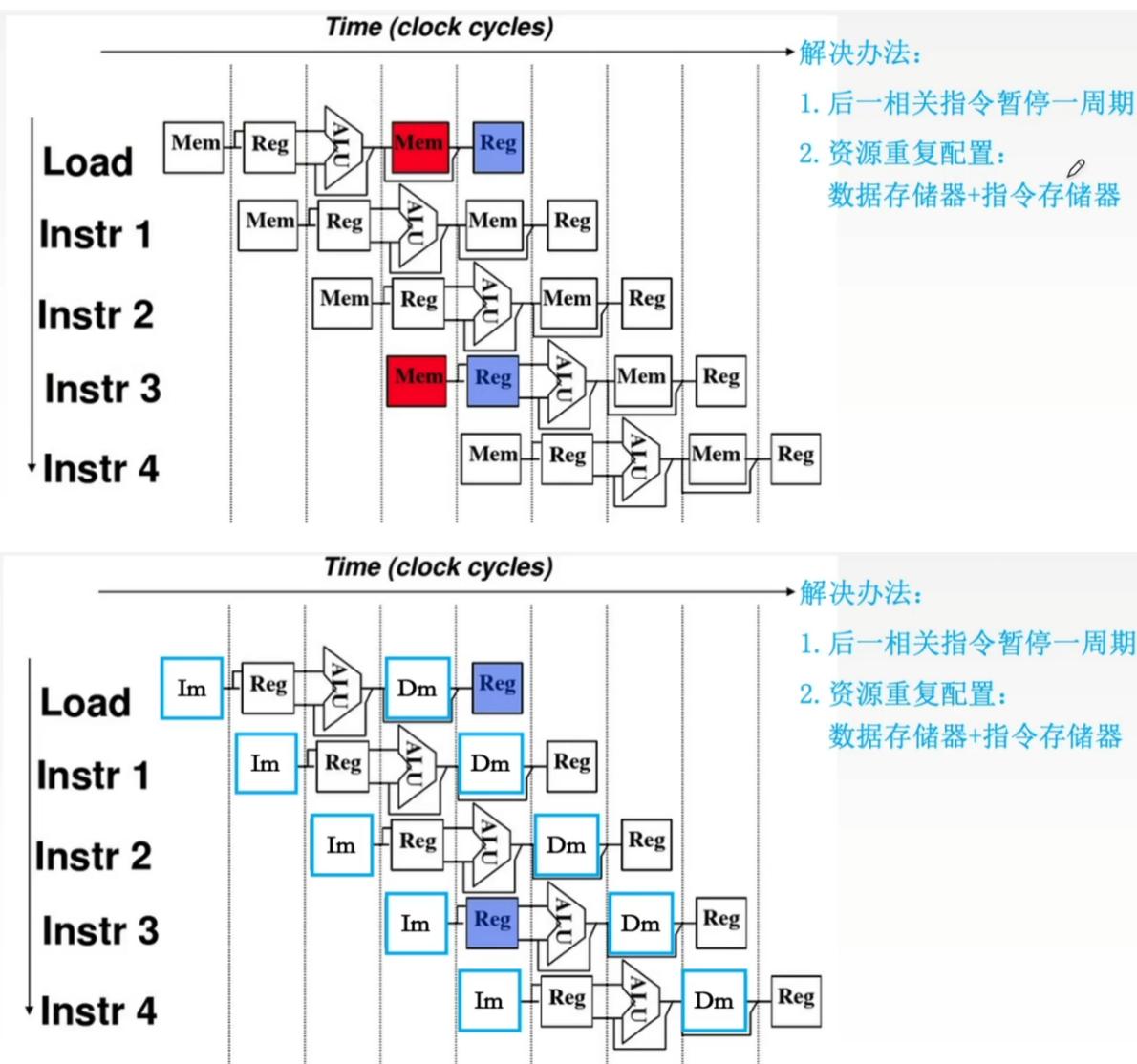
为方便流水线的设计，将每个阶段的耗时取成一样，以最长耗时为准。
即此处应将机器周期设置为100ns。

影响流水线的因素



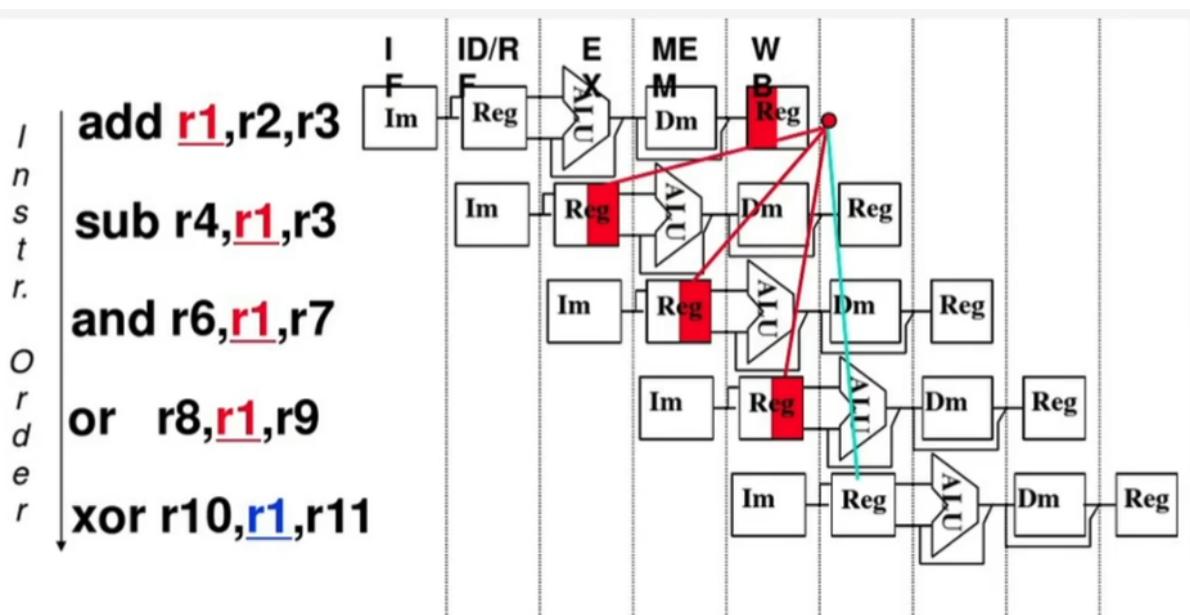
结构相关 (资源冲突)

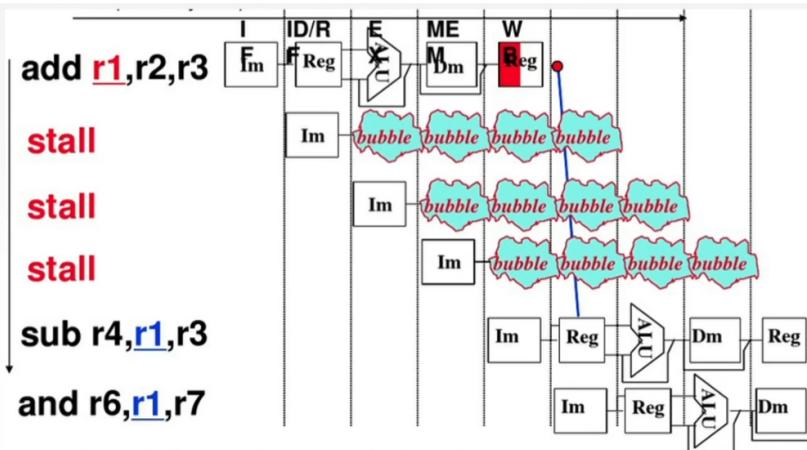
由于多条指令在同一时刻争用同一资源而形成的冲突称为结构相关。



数据相关 (数据冲突)

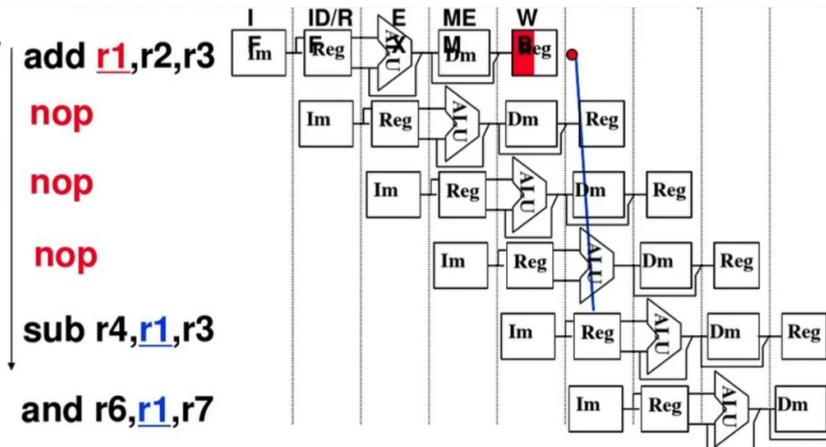
数据相关指在一个程序中，存在必须等前一条指令执行完才能执行后一条指令的情况，则这两条指令即为数据相关。





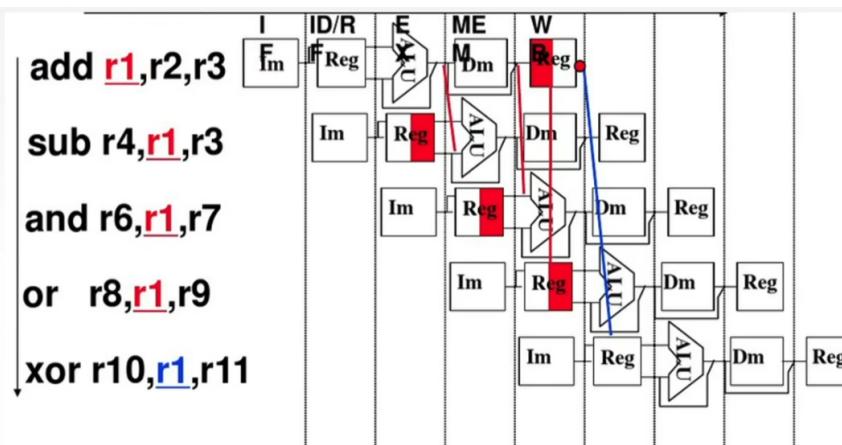
解决办法:

- 把遇到数据相关的指令及其后续指令都暂停一至几个时钟周期，直到数据相关问题消失后再继续执行。可分为硬件阻塞(stall)和软件插入“NOP”两种方法。



解决办法:

- 把遇到数据相关的指令及其后续指令都暂停一至几个时钟周期，直到数据相关问题消失后再继续执行。可分为硬件阻塞(stall)和软件插入“NOP”两种方法。

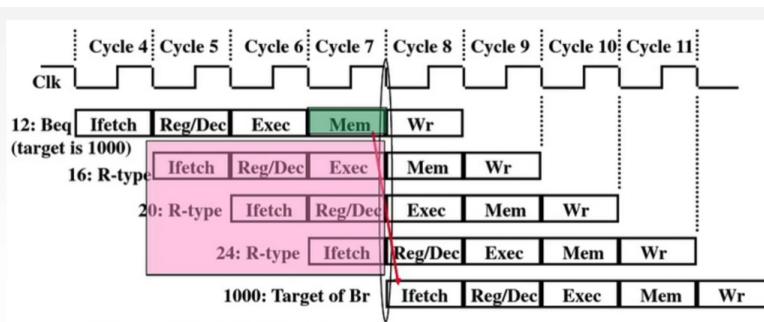


解决办法:

- 把遇到数据相关的指令及其后续指令都暂停一至几个时钟周期，直到数据相关问题消失后再继续执行。可分为硬件阻塞(stall)和软件插入“NOP”两种方法。
- 数据旁路技术。
- 编译优化：通过编译器调整指令顺序来解决数据相关。

控制相关 (控制冲突)

当流水线遇到转移指令和其他改变PC值得指令而造成断流时，会引起控制相关。



解决办法:

- 转移指令分支预测。简单预测（永远猜ture或false）、动态预测（根据历史情况动态调整）
- 预取转移成功和不成功两个控制流方向上的目标指令
- 加快和提前形成条件码
- 提高转移方向的猜准率

流水线的分类

部件功能级、处理机级和处理机间级流水线

根据流水线使用的级别的不同，流水线可分为部件功能级流水线、处理机级流水线和处理机间流水线。

部件功能级流水就是将复杂的算术逻辑运算组成流水线工作方式。例如，可将浮点加法操作分成求阶差、对阶、尾数相加以及结果规格化等4个子过程。

处理机级流水是把一条指令解释过程分成多个子过程，如前面提到的取指、译码、执行、访存及写回5个子过程。

处理机间流水是一种宏流水，其中每一个处理机完成某一专门任务，各个处理机所得到的结果需存放与下一个处理机所共享的存储器中。

单功能流水线和多功能流水线

按流水线可以完成的功能，流水线可分为单功能流水线和多功能流水线。

单功能流水线指只能实现一种固定的专门功能的流水线；

多功能流水线指通过各段间的不同连接方式可以同时或不同时地实现多种功能的流水线。

动态流水线和静态流水线

按同一时间内各段之间的连接方式，流水线可分为静态流水线和动态流水线。

静态流水线指在同一时间内，流水线的各段只能按同一种功能的连接方式工作。

动态流水线指在同一时间内，当某些段正在实现某种运算时，另一些段却正在进行另一种运算。这样对提高流水线的效率很有好处，但会使流水线控制变得很复杂。

线性流水线和非线性流水线

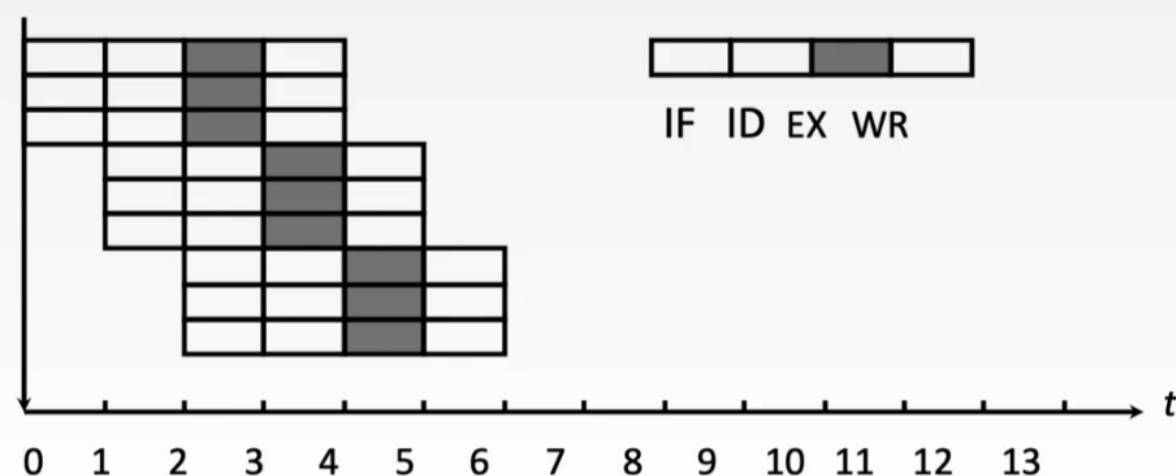
按流水线的各个功能段之间是否有反馈信号，流水线可分为线性流水线与非线性流水线。

线性流水线中，从输入到输出，每个功能段只允许经过一次，不存在反馈回路。

非线性流水线存在反馈回路，从输入到输出过程中，某些功能段将数次通过流水线，这种流水线适合进行线性递归的运算。

流水线的多发技术

超标量技术



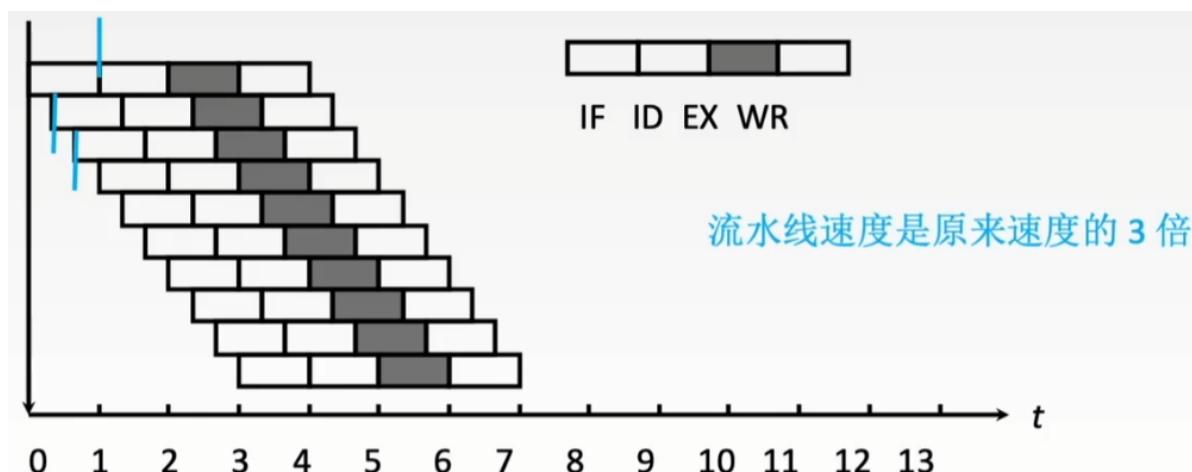
每个时钟周期内可并发多条独立指令

要配置多个功能部件

不能调整指令的执行顺序

通过编译优化技术，把可并行执行的指令搭配起来

超流水技术



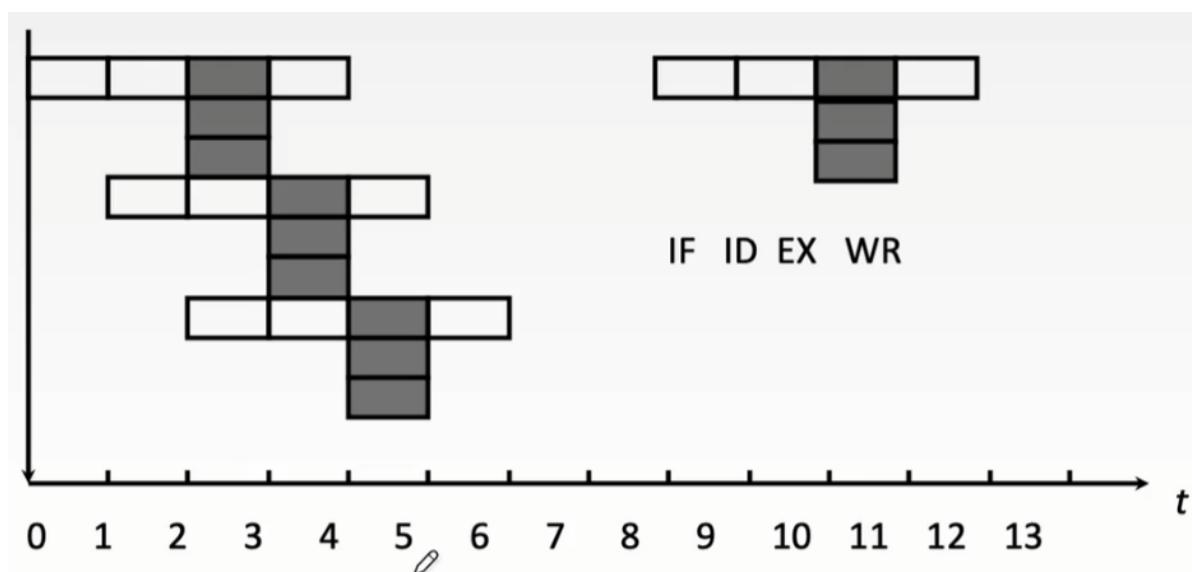
在一个时钟周期内 再分段（3 段）

在一个时钟周期内 一个功能部件使用多次（3 次）

不能调整 指令的 执行顺序

靠编译程序解决优化问题

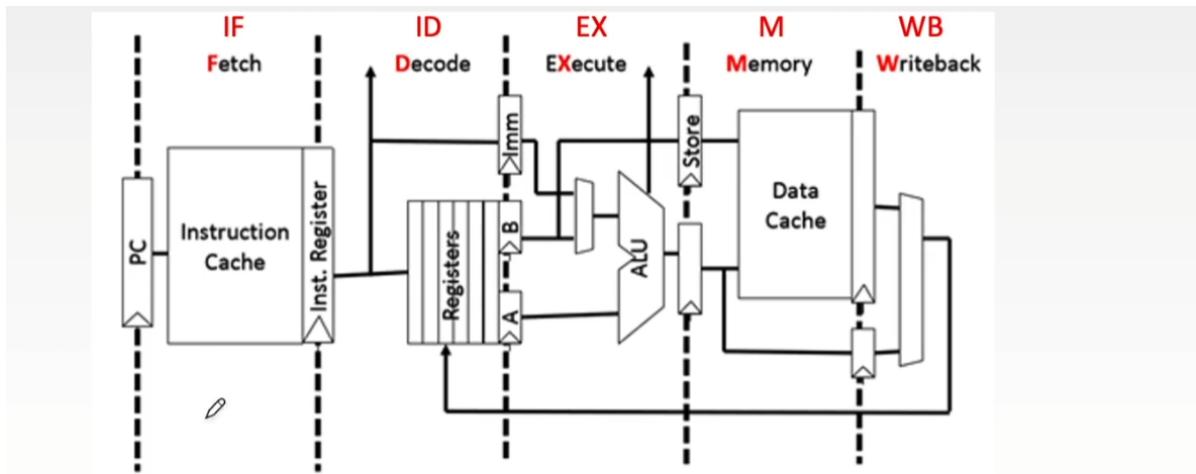
超长指令字



具有 多个操作码字段 的 超长指令字（可达几百位）

采用 多个处理部件

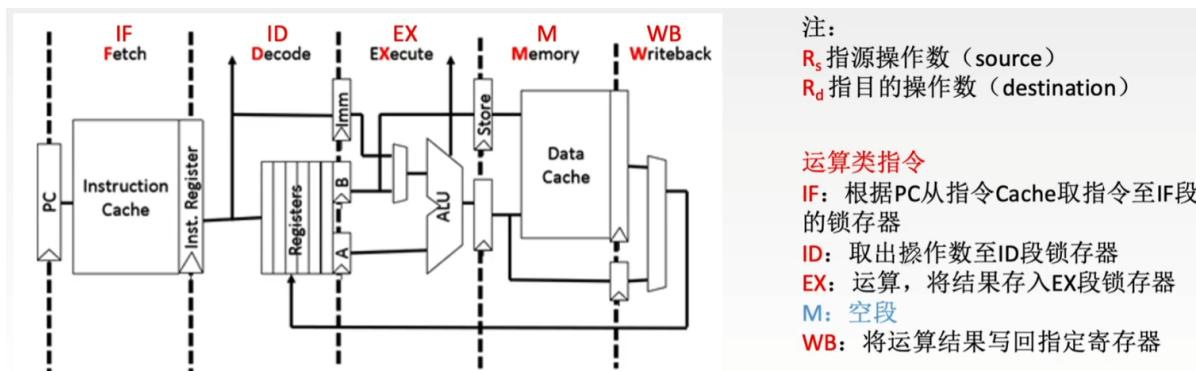
五段式指令流水线



①IF取指 → ②ID译码&取数 → ③EX 执行 → ④M访存 → ⑤WB写回寄存器

只有上一条指令进入ID段后，下一条指令才能开始IF段，否则会覆盖IF段锁存器的内容

运算类指令



①IF取指 → ②ID译码&取数 → ③EX 执行 → ④M访存 → ⑤WB写回寄存器

运算类指令举例

加法指令（两个寄存器相加）：
加法指令（寄存器与立即数相加）：
算数左移指令：

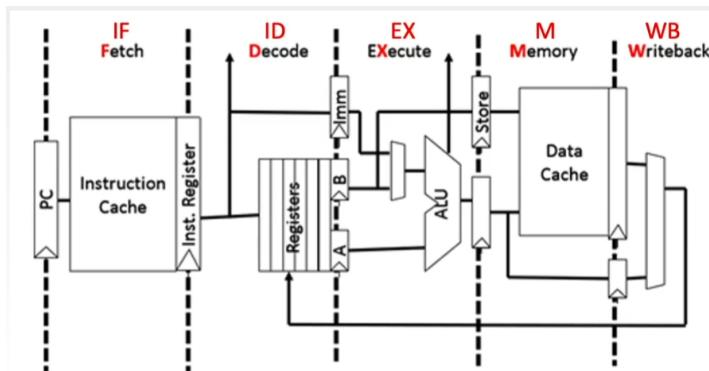
指令的汇编格式

ADD Rs,Rd
ADD #996,Rd
SHL Rd

功能

$(R_s)+(R_d) \rightarrow R_d$
 $996+(R_d) \rightarrow R_d$
 $(R_d) \lll 2 \rightarrow R_d$

LOAD指令



LOAD指令

IF: 根据PC从指令Cache取指令至IF段的锁存器

ID: 将基址寄存器的值放到锁存器A, 将偏移量的值放到Imm

EX: 运算, 得到有效地址

M: 从数据Cache中取数并放入锁存器

WB: 将取出的数写回寄存器

①IF取指 → ②ID译码&取数 → ③EX执行 → ④M访存 → ⑤WB写回寄存器

指令的汇编格式

LOAD Rd,996(Rs)

功能

(996+(Rs))→Rd

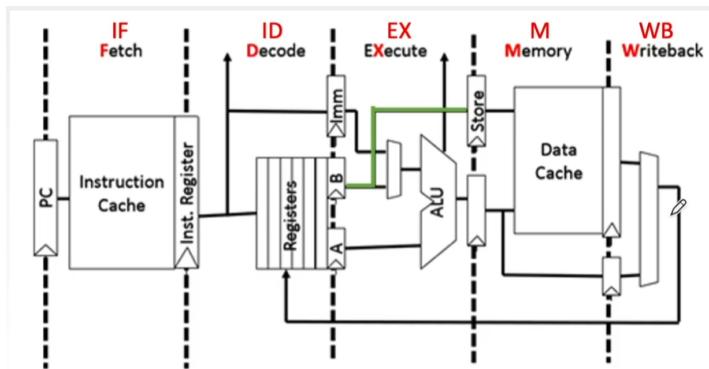
或简写为:

LOAD Rd,mem

通常, RISC处理器只有“取数LOAD”和
“存数STORE”指令才能访问主存

(mem)→Rd

STORE指令



STORE指令

IF: 根据PC从指令Cache取指令至IF段的锁存器

ID: 将基址寄存器的值放到锁存器A, 将偏移量的值放到Imm。将要存的数放到B

EX: 运算, 得到有效地址。并将锁存器B的内容放到锁存器 Store。

M: 写入数据Cache

WB: 空段

①IF取指 → ②ID译码&取数 → ③EX执行 → ④M访存 → ⑤WB写回寄存器

指令的汇编格式

STORE Rs,996(Rd)

功能

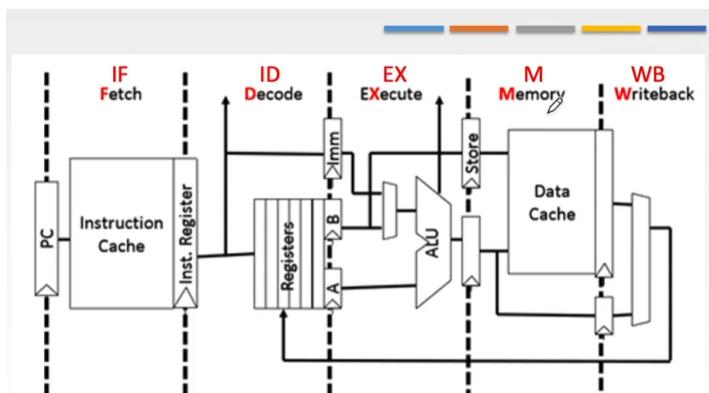
Rs→(996+(Rd))

或简写为:

STORE Rs,mem

Rs→(mem)

条件转移指令



转移类指令常采用相对寻址

条件转移指令

IF: 根据PC从指令Cache取指令至IF段的锁存器

ID: 进行比较的两个数放入锁存器A、B; 偏移量放入 Imm

EX: 运算, 比较两个数

M: 将目标PC值写回PC (左图没画全)

WB: 空段

很多教材把写回PC的功能段称为“WrPC段”, 其耗时比M段更短, 可安排在M段时间内完成

①IF取指 → ②ID译码&取数 → ③EX执行 → ④M访存 → ⑤WB写回寄存器

指令的汇编格式

beq Rs, Rt, #偏移量

功能

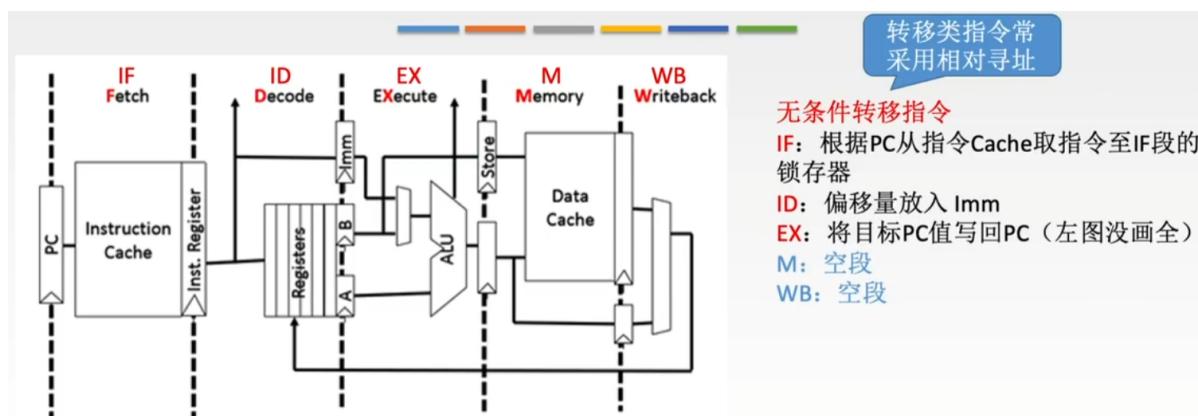
若 $(Rs) == (Rt)$, 则 $(PC) + \text{指令字长} + (\text{偏移量} \times \text{指令字长}) \rightarrow PC$; 否则 $(PC) + \text{指令字长} \rightarrow PC$

bne Rs, Rt, #偏移量

若 $(Rs) != (Rt)$, 则 $(PC) + \text{指令字长} + (\text{偏移量} \times \text{指令字长}) \rightarrow PC$; 否则 $(PC) + \text{指令字长} \rightarrow PC$

注: 通常在IF段结束之后PC就会自动+“1”

无条件转移指令



①IF取指 → ②ID译码&取数 → ③EX执行 → ④M访存 → ⑤WB写回寄存器

指令的汇编格式

jmp #偏移量

功能

(PC)+指令字长+(偏移量×指令字长)→PC

无条件转移指令

IF: 根据PC从指令Cache取指令至IF段的锁存器

ID: 偏移量放入 Imm

EX: 将目标PC值写回PC (左图没画全)

M: 空段

WB: 空段

“WrPC段”耗时比EX段更短，可安排在EX段时间内完成。WrPC段越早完成，就越能避免控制冲突。当然，也有的地方会在WB段时间内才修改PC的值

例题. 假设某指令流水线采用“按序发射，按序完成”方式，没有采用转发技术处理数据相关，并且同一寄存器的读和写操作不能在同一个时钟周期内进行。若高级语言程序中某赋值语句为 $x = a + b$ ， x 、 a 和 b 均为int型变量，它们的存储单元地址分别表示为 $[x]$ 、 $[a]$ 和 $[b]$ 。该语句对应的指令序列及其在指令流中的执行过程如下图所示。

```
I1 LOAD R1, [a] M[a] -> R1
I2 LOAD R2, [b] M[b] -> R2
I3 ADD R1, R2 (R1) + (R2) -> R2
I4 STORE R2, [x] (R2) -> M[x]
```

则这4条指令执行过程中I3的ID段和I4的IF段被阻塞的原因各是什么？

I3与I1和I2存在数据相关；

I4的IF段必须在I3进入ID段后才能开始，否则会覆盖IF段锁存器的内容

指令	时间单元													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
I ₁	IF	ID	EX	M	WB									
I ₂		IF	ID	EX	M	WB								
I ₃			IF				ID	EX	M	WB				
I ₄							IF				ID	EX	M	WB