

# 第三章 流水线技术

## ※本章目标

理解指令流水线的组成&工作原理，掌握流水线冒险的处理技术

## ※主要内容

### (1) 流水线的概念

工作原理，组成要求，分类，性能指标

### (2) 流水线的冒险处理

流水线的基本组成，

相关与冒险，结构冒险处理，数据冒险处理，控制冒险处理

### (3) 流水线的实现

流水线数据通路实现，流水线控制器的实现

# 第1节 流水线的概念

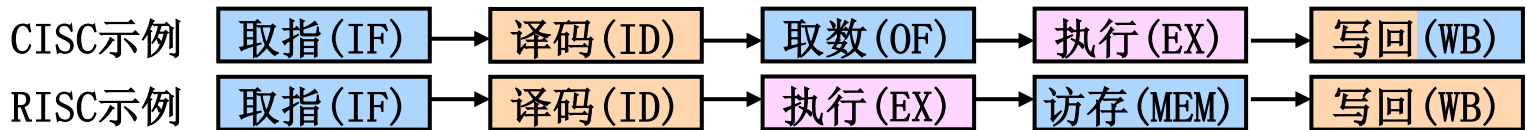
※主要内容：工作原理，组成要求，分类，性能指标，段数选择

## 1、流水线 (pipeline) 的工作原理

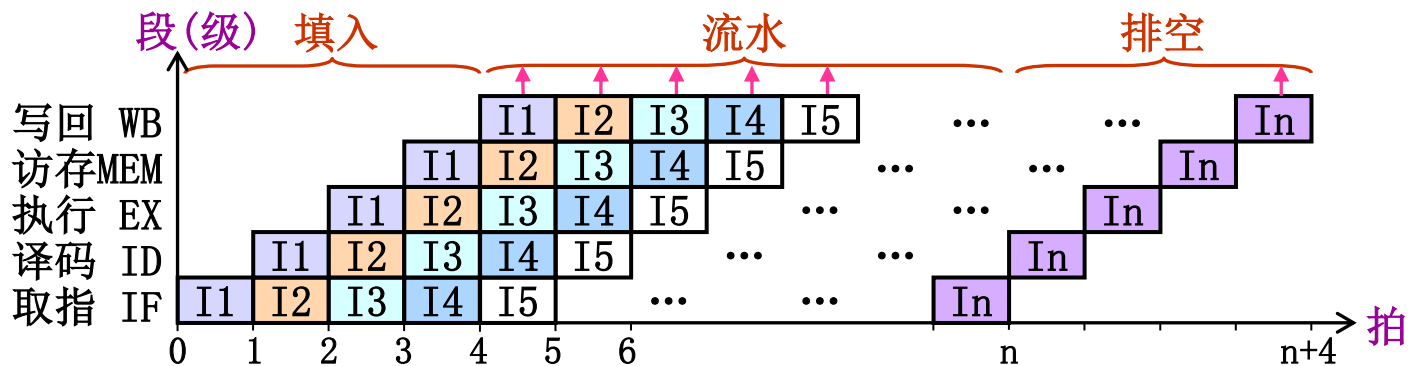
\*基本思想：指令执行过程分为多个阶段， ←基础 (分割)

各个阶段可同时处理不同指令的操作 ←效果 (重叠)

\*基本组成：所有功能段按序连接 ←类似多周期



\*工作原理：每条指令按序通过各个段，不同指令执行过程重叠



程序执行时间— $T_{流水} = m\Delta t + (n - 1)\Delta t$  ,  $T_{串行} = n \cdot (m\Delta t)$

## 2、流水线组成的基本要求

**\*要求1：各个段的操作相互独立**

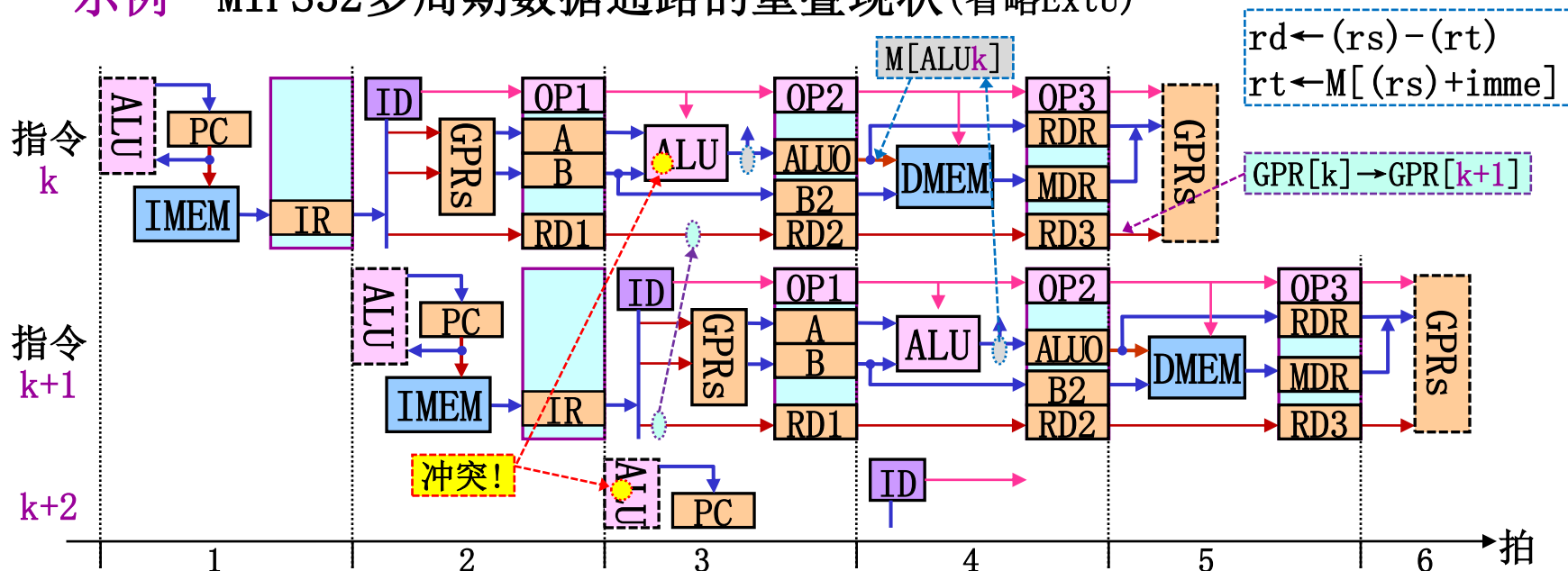
方法一①部件不能复用 ②操作的数据放在时序部件中(结果也如此)

实现一各个段的部件仅本段使用，各个段增设段间寄存器(末段除外)

思考：哪些信息需保存到段间REG？在哪些段保存？如何实现段结束时保存？

数据/地址/命令，产生→使用间所有段，采用边沿触发方式

示例—MIPS32多周期数据通路的重叠现状(省略ExtU)



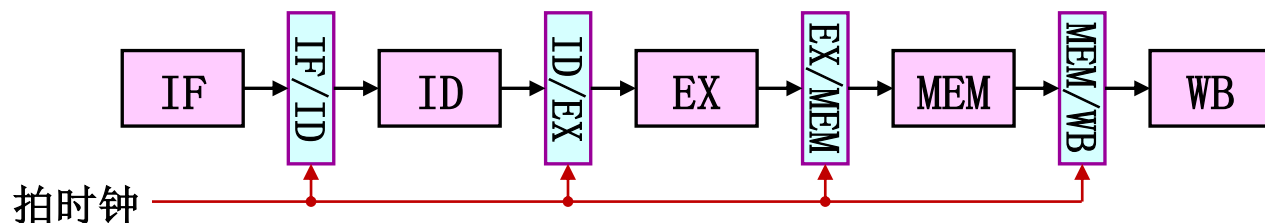
思考：数据/地址/命令，产生-使用之间的所有段，寄存器采用边沿触发

## \*要求2: 各流水段的操作同步

←连续重叠所需

实现—设置公共时钟，同时写段间寄存器(即写结果同步)

$L \leftarrow \text{拍时钟周期} = \max\{\text{各段延迟}\}$



## \*要求3: 各流水段的操作无冲突

←保证结果正确

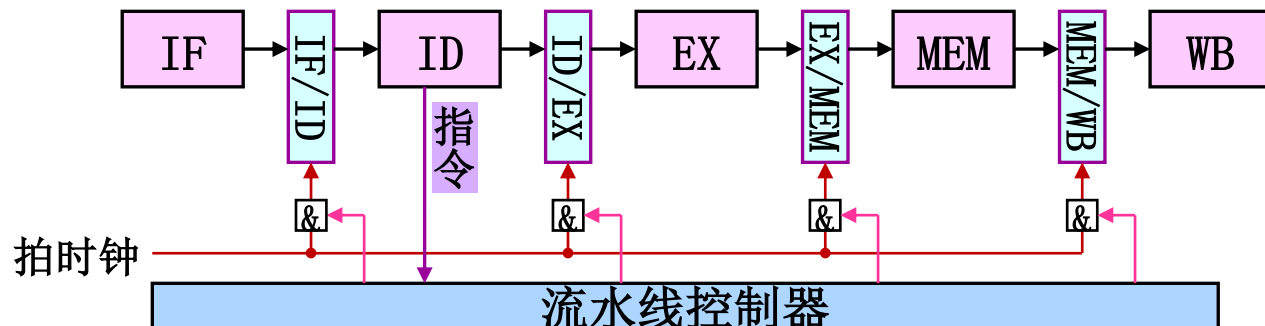
冒险(冲突)—指流水线因某些原因无法正确执行后续指令的现象

冒险类型—结构冒险、数据冒险、控制冒险

(如部件复用) (源-目OPD相关) (分支指令)

I1:  $R2 \leftarrow (R1) + 5$   
I2:  $(R2) = 6$ 时  $PC \leftarrow I1$   
I3:  $R3 \leftarrow (R1) + 5$

实现—增设部件及控制机制，解决冒险(如阻塞后续指令)

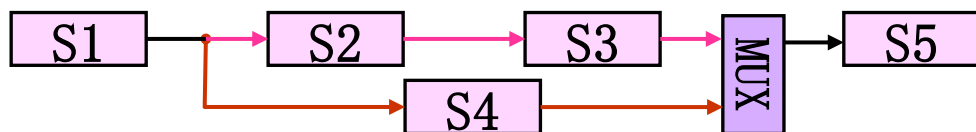


### 3、流水线的分类

——即属性

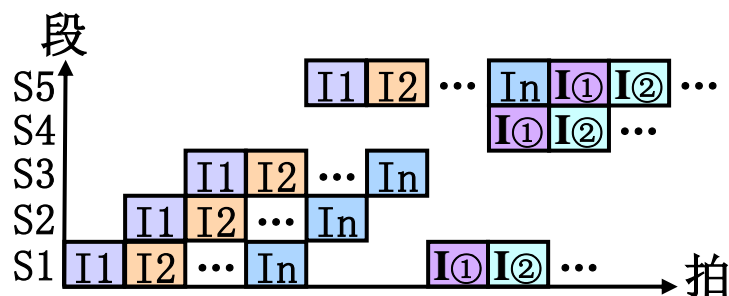
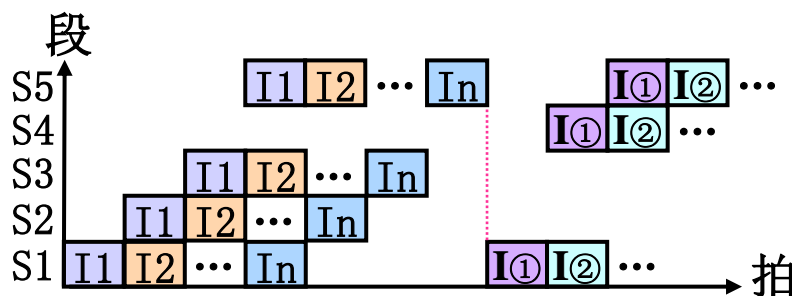
**\*按处理级别分类：**操作级、指令级、处理机级(宏流水)

**\*按功能分类：**单功能流水线、多功能流水线

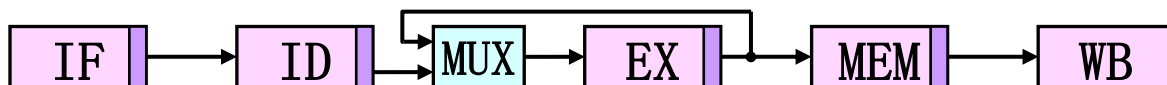


动态→多功能  
单功能→静态

**\*按工作方式分类：**静态流水线、动态流水线



**\*按结构分类：**线性流水线、非线性流水线(复用部件)



**\*按流入/流出次序分类：**顺序流水线、乱序流水线

示例—理想流水线为线性、动态、按序流动、标量流水线

## 4、流水线的性能指标

**\*吞吐率 (Through Put):**  $TP = \text{任务数}n / \text{处理}n\text{个任务总时间}$

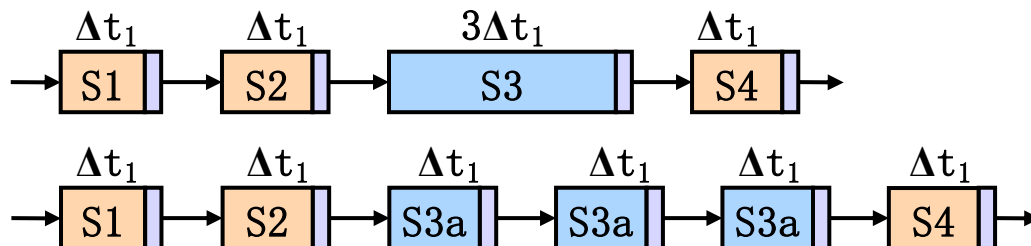
实际吞吐率—  $TP = \frac{n}{T_{\text{流水}}} = \frac{n}{m\Delta t + (n-1)\Delta t}$  ,  $\Delta t = \max\{\Delta t_i\}$

最大吞吐率— 当  $n \gg m$  时,  $TP_{\max} = 1/\Delta t$ , 即拍长的倒数

优化吞吐率的方法— 减少  $\Delta t$  (即消除瓶颈段)

瓶颈段处理方案:

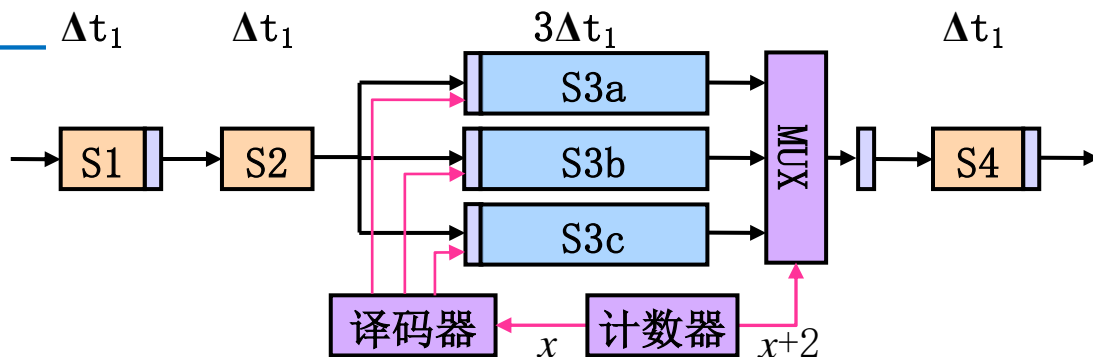
① 细分瓶颈段—



② 重复设置部件—

(轮流使用)

(控制方法)



**\*加速比 (Speedup):**  $S = \text{串行执行时间} / \text{流水执行时间}$

实际加速比—  $S = \frac{n \cdot m \Delta t}{m \Delta t + (n-1) \Delta t} = \frac{nm}{m+n-1}$

最大加速比—当  $n \gg m$  时,  $S_{\max} = m$ , 即流水线段数

优化加速比的方法—增加段数  $m$  (可减小  $\Delta t$ )

**\*效率 (Efficiency):** 即部件利用率,  $E = \text{部件使用时间} / \text{流水执行总时间}$   
= 任务所占时空区 /  $m$  个段总时空区

实际效率—  $E = \frac{e_1 + e_2 + \dots + e_m}{m} = \frac{m \cdot ((n \Delta t) / ((m+n-1) \Delta t))}{m} = \frac{n}{m+n-1}$

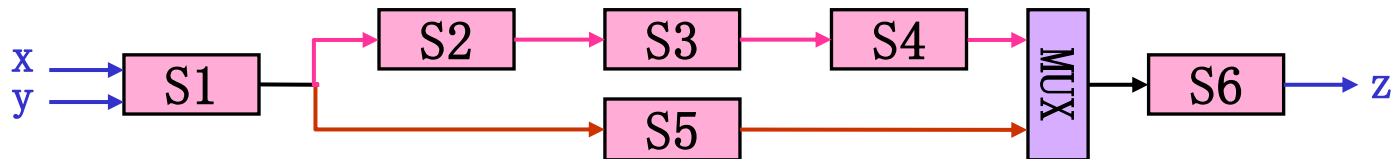
对于线性流水线,  $E = S / m = TP \cdot \Delta t$

最高效率—当  $n \gg m$  时,  $E_{\max} = 1$

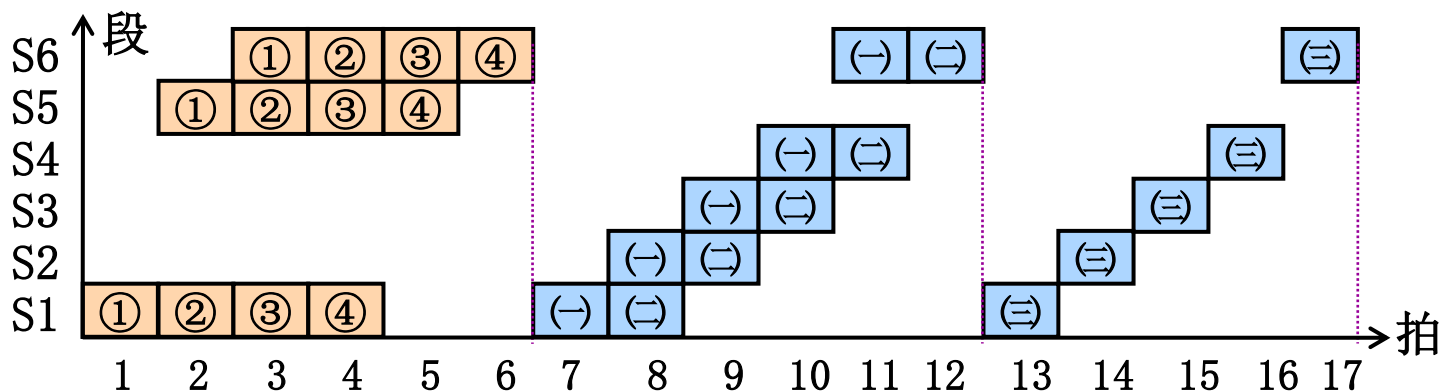
优化效率的方法—增加任务数量  $n$



**例：**静态、加/乘双功能流水线中，加法功能仅使用3个段，计算并分析实现  $\Pi(a_i + b_i)$  的性能，其中  $i=1\sim 4$ 。



**算法一** 先完成所有加法、再做乘法(减少功能切换)



**性能一**  $TP=7/(17\Delta t)$ ,  $S=1.588$ ,  $E=26.5\%$

**分析一** ①指令数少，填入、排空时间较多

←仅拍3和4流水

②静态流水，排空后才能切换功能

←(-)的S1

③数据冒险，消除(如阻塞)后才继续流水

←(≡)的S1

## 第2节 流水线的冒险处理

※主要内容：流水线基本组成，相关与冒险，结构/数据/控制冒险处理

### 1、流水线的基本组成

#### (1) 单周期数据通路的实现 (以MIPS32为例)

指令周期—=1个  $T_c = \max \{ \text{各指令全部操作时延} \}$

\*指令系统分析：(仅讨论操作需求，忽略部件参数需求)

数据操作—32b整数加/减/或运算，零/符号扩展，  
GPRs读/写 ( $\leq [2\text{次R}+1\text{次W}] / \text{指令}$ )，MEM存/取

指令寻址操作—32b整数加等  $\leftarrow PC+4$  及相对寻址

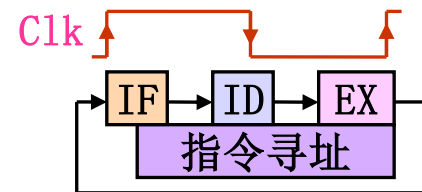
\*部件组织：(基于指令执行过程+指令功能，部件不能复用)

指令部件—设置PC，缺省IR(减少时序逻辑操作数)

$\leftarrow \text{IMEM} + \text{写IR} + \text{读MEM} + \text{写GPR} \leq 3\text{次}$

运算部件—设置ALU及ExtU(器件无法复用)，设置ACU

其他部件—GPRs (2个R端口+1个W端口)，IMEM+DMEM

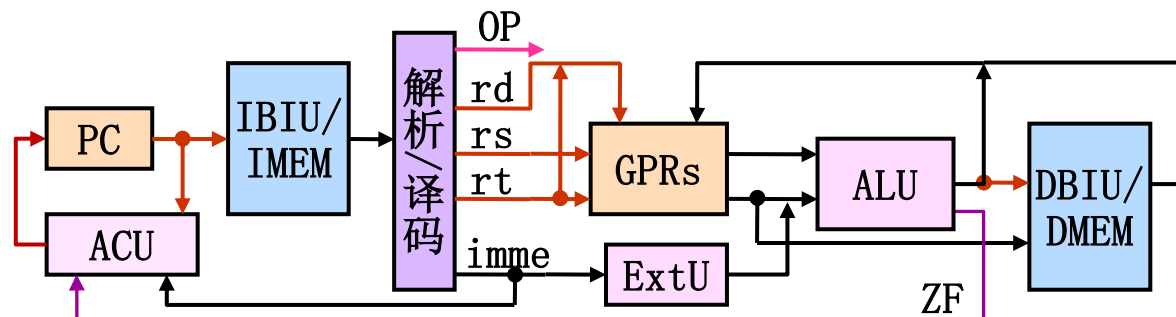


## \*部件互连组织：（基于指令执行过程+各指令功能）

指令执行过程—取指+译码+执行（如读GPRs+ALU+[读DMEM+]写GPRs）

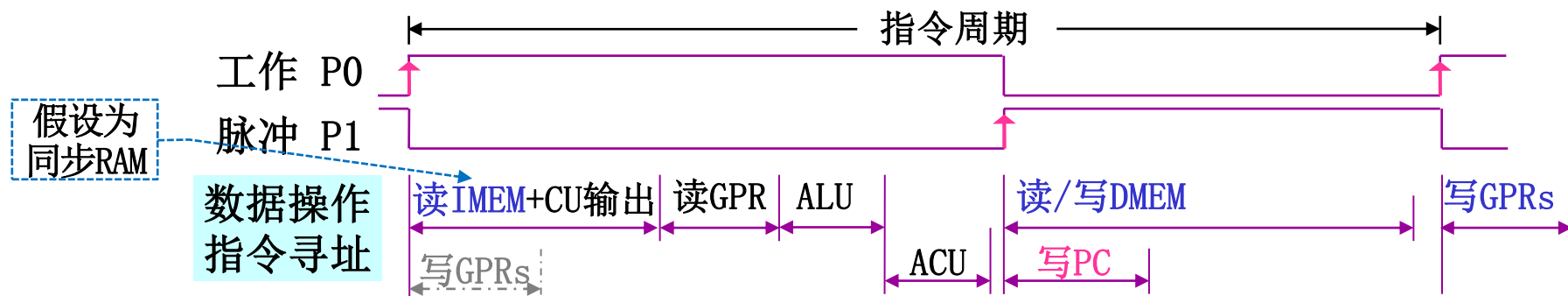
部件互连—专用结构（单周期所需），逐条增加（部件/路径）

add	$rd \leftarrow (rs) + (rt)$
ori	$rt \leftarrow (rs) \mid ZExt(imme)$
lw	$rt \leftarrow M[(rs) + SExt(imme)]$
sw	$M[(rs) + SExt(imme)] \leftarrow (rt)$
beq	if $((rs) = (rt))$ $PC \leftarrow (PC) + 4 + SExt(imme) \ll 2$



操作时序—根据指令执行过程&功能组织（关注时序逻辑操作）

← 基于最复杂指令



**思考：**结束时写GPRs会影响下条指令的执行吗？ACU操作为何在ALU之后？

思考：不影响， $T_{写GPRs} < T_{取指} + T_{译码}$ ；beq的ACU输出依赖于ALU结果

## (2) 多周期数据通路的实现

指令周期 =  $n$  个  $T_c$ ,  $T_c = \max \{ \text{各基本} \mu\text{OP 时长} \}$

ALU/GPRs等  $\rightarrow \perp \leftarrow$  不考虑长时延  $\mu\text{OP}$

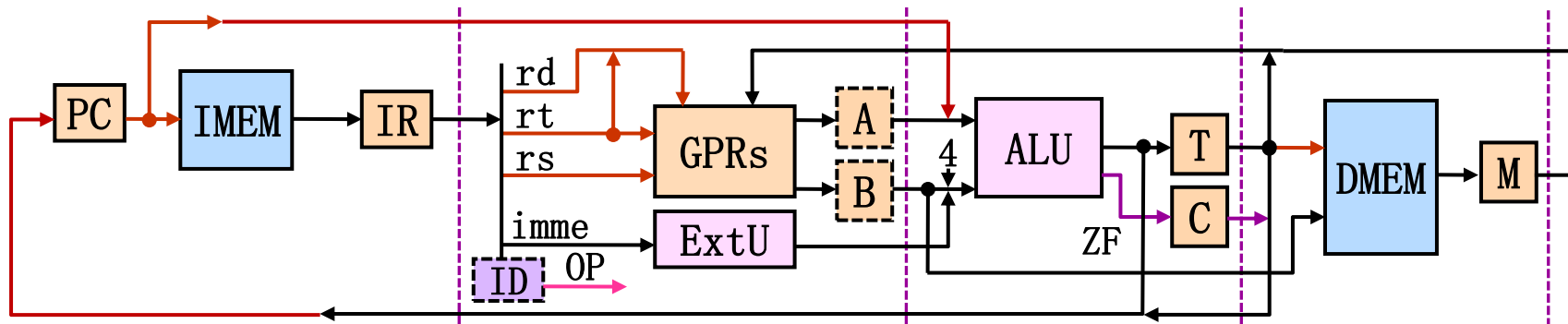
部件组织 — 基于单周期DP, 进行部件调整 (基于复用方案);

确定  $T_c$  长度, 划分功能段 (可占多个  $T_c$ );



增设附加REG (在各复用段末尾处)  $\leftarrow$  部件复用/简化控制

互连组织 — 基于各段功能, 按指令需求连接 (类似于单周期)



**思考:** (PC)+4何时实现? beq指令用ALU实现相等判断&相对寻址, 何时进行?

思考: 取指时; 读GPRs段实现  $T \leftarrow (PC) + \text{SExt}(\text{imme})$ 、EX段实现  $C \leftarrow (A) = (B)$ ?

## 指令执行过程组织— (可有多种方案[如beq在EX段写结果])

	add: $rd \leftarrow (rs) + (rt)$ ori: $rt \leftarrow (rs) \mid \text{imme}$	lw: $rt \leftarrow M[(rs) + \text{imme}]$ sw: $M[(rs) + \text{imme}] \leftarrow (rt)$	if: $((rs) = (rt))$ $PC \leftarrow (PC) + 4 + \text{imme} \ll 2$
IF	$IR \leftarrow IM[(PC)]$ ,		$PC \leftarrow (PC) + 4$
ID	$A \leftarrow R[rs], B \leftarrow R[rt]$ ,		$T \leftarrow PC + \text{ExtU}$
EX	$T \leftarrow (A) \text{ op } (B) \text{ 或 } \text{ExtU}$	$T \leftarrow (A) + \text{ExtU}$	$C \leftarrow ((A) = (B)) ? 1 : 0$
MEM	$R[rd] \leftarrow (T)$	$M \leftarrow DM[T]$	$DM[T] \leftarrow (B)$
WB		$R[rt] \leftarrow (M)$	$PC \leftarrow T (C=1 \text{ 时})$

分时复用ALU

※多周期→流水线需解决的问题:

操作相互独立— 结果隔段使用 (增设附加REG暂存结果[如EX段暂存B])

操作同步— 各段时延可能不等 (用同一时钟信号控制结果暂存)

操作无冲突— 部件复用 (增设部件[如IF段加法器]),

写GPR时间不同 (使时间唯一[如add移至WB段]),

写PC有2次 (仅在IF写[选择数据])

I1:lw	IF	ID	EX	MEM	WB
I2:add	IF	ID	EX	WB	

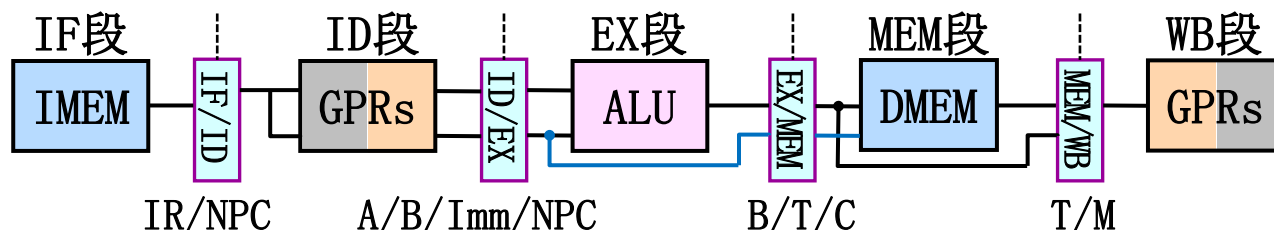
### (3) 流水线的基本组成

**\*各个段功能组织：基于多周期** (部件在同一个段使用/写PC仅在IF段)

段 \ 指令	add: $rd \leftarrow (rs) + (rt)$ ori: $rt \leftarrow (rs) \mid \text{imme}$	lw: $rt \leftarrow M[(rs) + \text{imme}]$ sw: $M[(rs) + \text{imme}] \leftarrow (rt)$	if: $((rs) = (rt))$ $PC \leftarrow (PC) + 4 + \text{imme} \ll 2$
IF	$IR \leftarrow IM[(PC)]$ , $NPC \leftarrow (PC) + 4$ , $PC \leftarrow NPC$		
ID	$A \leftarrow R[rs]$ , $B \leftarrow R[rt]$ , $Imm \leftarrow \text{ExtU}(\text{imme})$		
EX	$T \leftarrow (A) \text{ op } (B) \text{ 或 } Imm$	$T \leftarrow (A) + Imm$	$T \leftarrow (NPC) + Imm$ $C \leftarrow ((A) = (B)) ? 1 : 0$
MEM	空闲	$M \leftarrow DM[T]$	$DM[T] \leftarrow (B)$ $PC \leftarrow T (C=1 \text{ 时})$
WB	$R[rd] \leftarrow (T)$	$R[rt] \leftarrow (M)$	

**\*基本组成：增设部件** (IF段加法器、EX段比较器)，

增设段间REG，段间REG写入用统一时钟信号控制

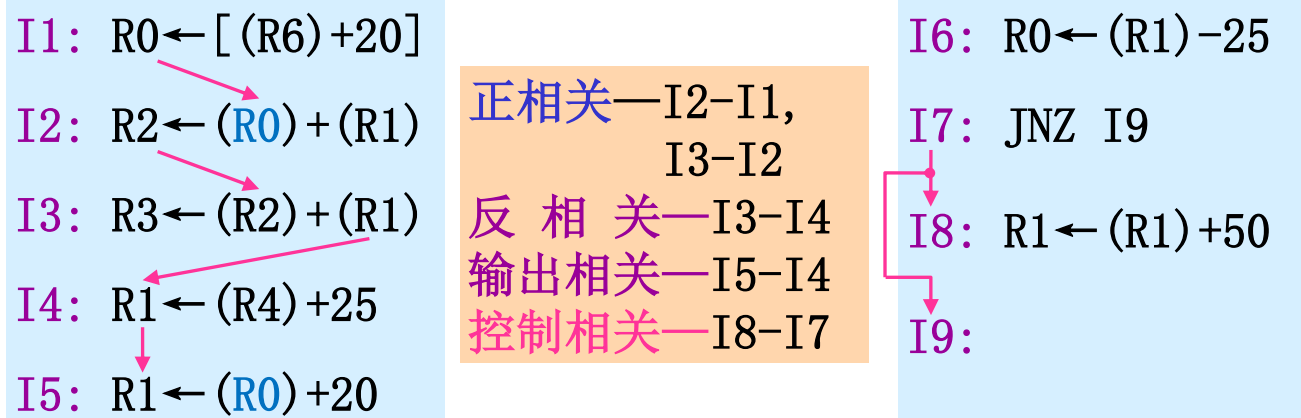


**思考：** 仅在IF段写PC何时实现？  $PC \leftarrow EX/MEM. C ? EX/MEM. T : NPC$

## 2、相关与冒险

**\*相关 (dependence):** 指令间存在的依赖关系

**类型一** 数据相关(正相关)、名相关(反相关/输出相关)、控制相关



**\*冒险 (Hazard, 冲突):** 因相关等原因引起的流水异常现象

**冒险类型一** 结构冒险、数据冒险、控制冒险

可能需停顿

└←无关于相关    └←相关距离 < 读-写间隔时 (正相关≠冒险)

**思考:** 上页流水线的GPR读-写间隔为几拍?

3拍

**程序执行时间一**  $T_{\text{流水}} = T_{\text{理想流水}} + T_{\text{结构停顿}} + T_{\text{数据停顿}} + T_{\text{控制停顿}}$

### 3、结构冒险处理

**\*产生原因：** 争用硬件资源(部件或路径)

如：MEM采用冯·诺依曼结构、(PC)+4复用ALU等

**\*处理方法：**

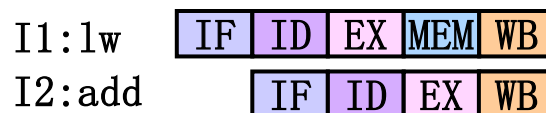
重复设置资源— 消除资源争用，适于高频率冲突

① 部件不复用、使用时间唯一

如：MEM采用哈佛结构、PC+4增设地址加法器

又如：IF段及MEM段的写PC，可合并到IF段(选择不同地址)

再如：写GPRs统一安排在第5个段



② 部件互连采用专用结构

分时使用资源— 避免资源争用，适于低频率冲突

如：冲突时使流水线停顿(后续指令暂停)

**\*应用选择：** 权衡性能损失～硬件成本



### 示例1—CISC流水线的访存冲突处理 (Intel PII采用)

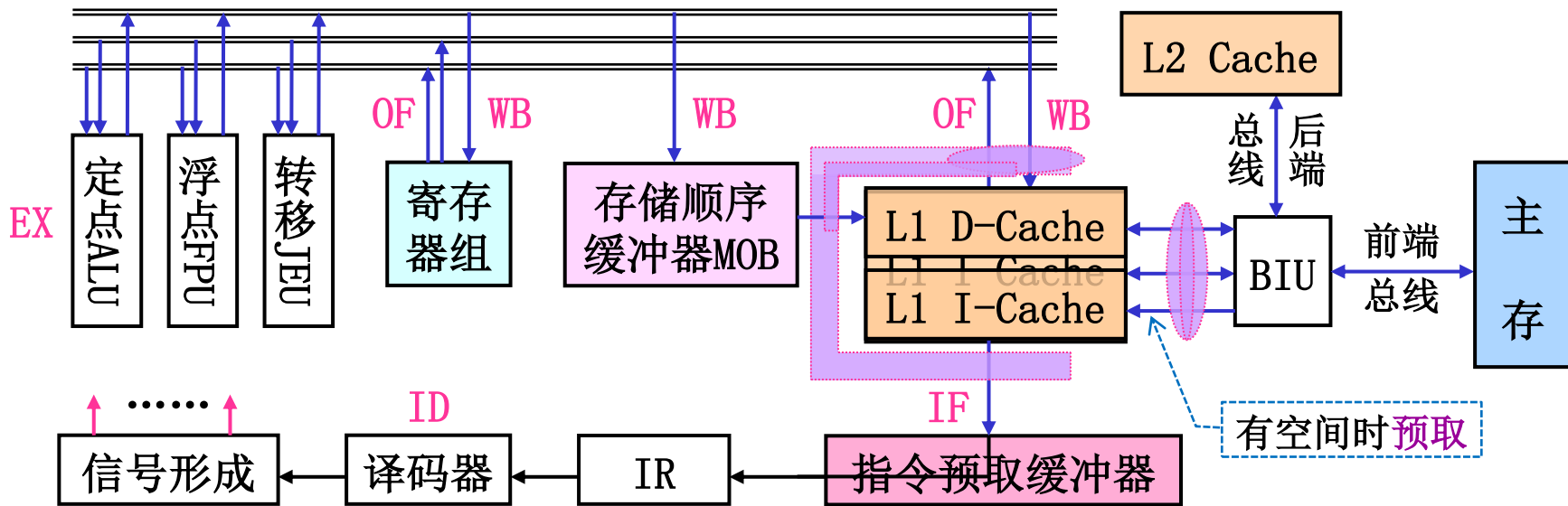
## 流水线组成—IF+ID+OF+EX+WB

### ←适于R-M型指令集

## 处理方案一（策略：降低冲突概率→消除冲突）

MOB可在L1-D\$闲时写

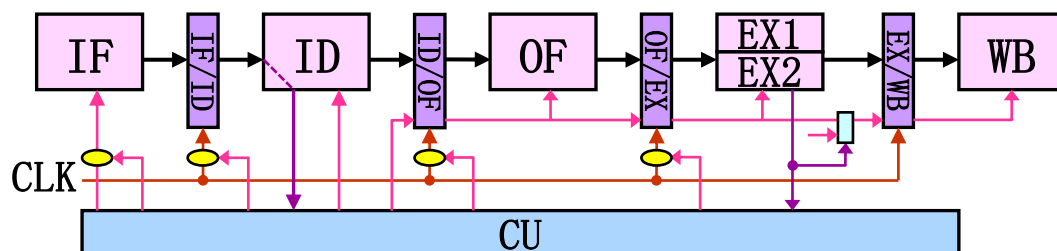
- ①增设一个MEM (IF-OF/WB冲突 ↓ [仅L1-D\$与L1-I\$同时缺失时])
- ②增设数据MOB (OF-WB冲突 ↓ [仅MOB及OF段同时访问L1\$时])
- ③增设指令预取缓冲器 (L1-D\$与L1-I\$冲突 ↓ [仅同时缺失时])
- ④L1-D\$与L1-I\$同时缺失时串行访问L2\$/主存 (解决冲突)



示例2—流水线的功能段不等长处理 (效果: 拍时钟从 $T_{\max}$ 段 $\rightarrow T_{\text{基本段}}$ )

流水线组成—ID段指令译码 (基于IF/ID. IR), EX段不等长

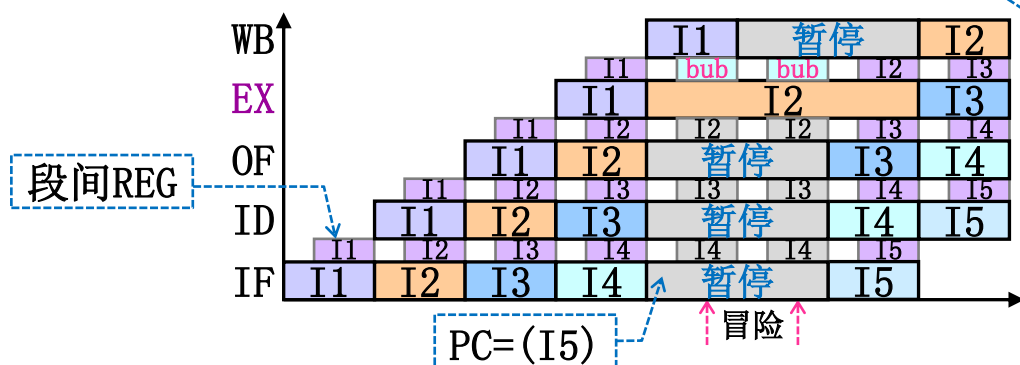
操作控制: ID后各段源自段间REG. Cmd, IF&ID段源自CU



处理方案—除EX外的段暂停, 直到冒险消除 (EX段完成)

暂停方法: EX之前段不写段间REG及状态部件, ←锁步机制1  
 $\hookrightarrow$  段状态不变  $\hookrightarrow$  如PC不变 (后能正确取指)

EX段产生气泡 (即空操作uOPCmd) ←锁步机制2  
 $\hookrightarrow$  写入段间REG. Cmd (WB段的控制源)



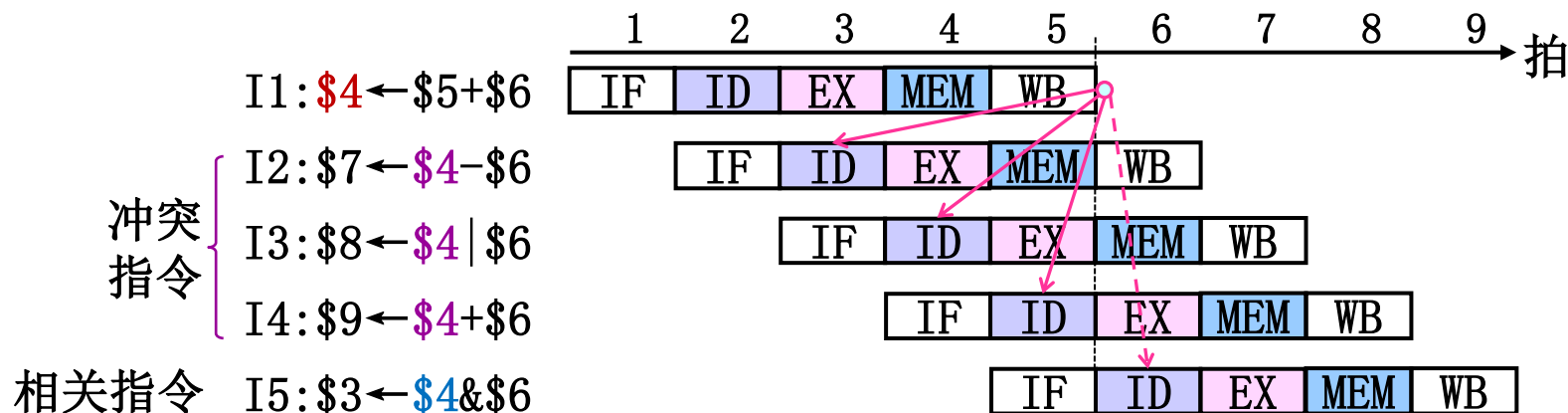
即nop指令功能 (不改变  
MEM/GPRs/PSR的值)

**思考:** 如何获知EX段的时长?  
各部件时延固定, 基于当前uOPCmd

## 4、数据冒险处理 (以MIPS为例)

**\*产生原因：** 因重叠执行，指令所需数据不可用

**\*冒险类型：** 写后读 (Read After Write, RAW) 冒险等



**\*处理方法：** 阻塞法、转发法、乱序执行法

**场景一**食堂打饭时，甲需要用乙(乙排在甲之前)的饭卡，处理方法？

甲	甲后面的人	打饭窗口状态
处理一①原地等&拿卡后再走	一起等	有停顿(原次序)
②往前走&途中拿卡	跟着走	无停顿(原次序)
③原地等&拿卡后再走	绕着走	无停顿(乱次序)

**\*阻塞法：**冲突指令及后续指令暂停，直到冒险消除（数据就绪）

冒险消除时要求— IF/ID(段间REG)及PC不变，其余段间REG为气泡

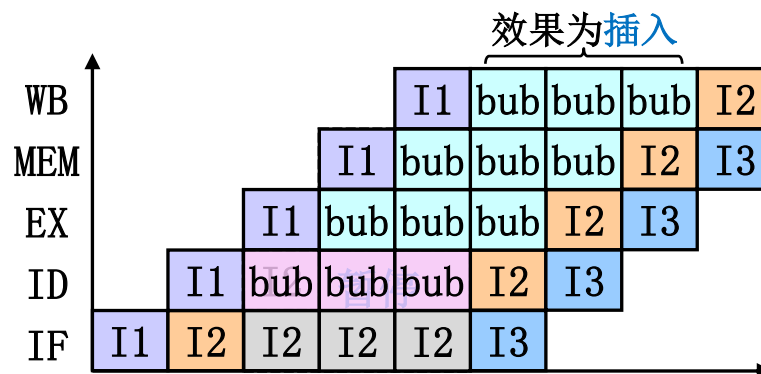
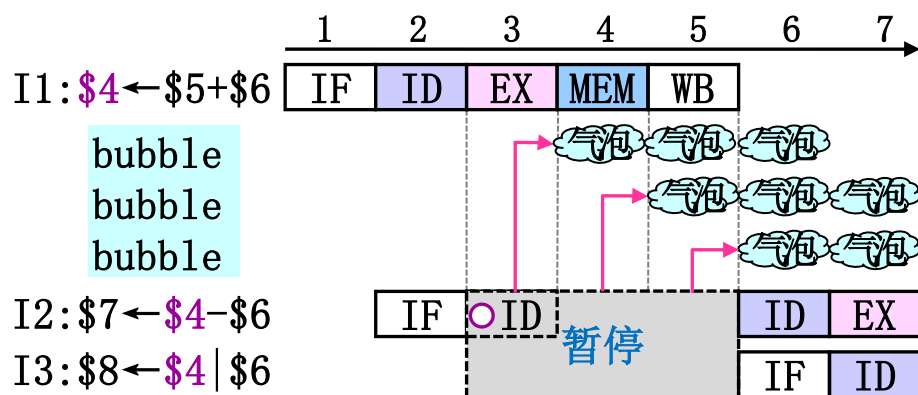
暂停方法— ID段产生气泡（空操作u0PCmd），

←锁步机制2

后续段的控制源

IF段暂停（不写IF/ID及PC）

←锁步机制1



思考①：ID产生气泡，IF不暂停行吗？

RAW消除时，PC不是I3、IF/ID. IR不是I2

思考②：为何不在IF产生气泡？

RAW消除时，IF/ID. IR为nop指令，I2无法执行

思考③：ID段产生气泡的方法？

用MUX选择Cmd(当前/nop指令)，写ID/EX. Cmd

**停顿拍数— 数据欲读-可读的相差拍数**

思考④：如何得到停顿拍数？

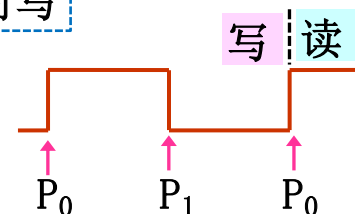
ID段每拍比较 (IF/ID. IR. Rs ~ 后续段间REG. Rd)，直至匹配失败（冒险消除）

（阻塞法停顿拍数由CU实现）

**例1:** MIPS流水线中, 写GPRs放在后半拍完成(WB的下一拍才能读出所写数据)。现有如下MIPS指令序列:

I1: add \$4, \$5, \$6 ; \$4 ← \$5 + \$6  
 I2: sub \$7, \$4, \$6 ; \$7 ← \$4 - \$6  
 I3: or \$8, \$4, \$6 ; \$8 ← \$4 | \$6  
 I4: sw \$6, 20(\$4) ; M[\$4 + 20] ← \$6  
 I5: lw \$9, 20(\$8) ; \$9 ← M[\$8 + 20]

P<sub>0</sub>时写



问: (1)哪些指令之间存在RAW冒险?

冲突指令放在前面

(2)采用阻塞法处理RAW冒险时, 指令序列执行时间?

**解:** (1)RAW冒险有: I2-I1、I3-I1、I4-I1, I5-I3

(2)I2-I1冒险需停 **3** 拍, I3-I1、I4-I1冒险各需停 **0** 拍;

I5-I3冒险需停 **2** 拍;

随I2-I1自动消除

执行时间 =  $T_{\text{理想流水}} + T_{\text{冒险停顿}}$

$$= [5t + (5-1)t] + (3+2)t = 14t$$

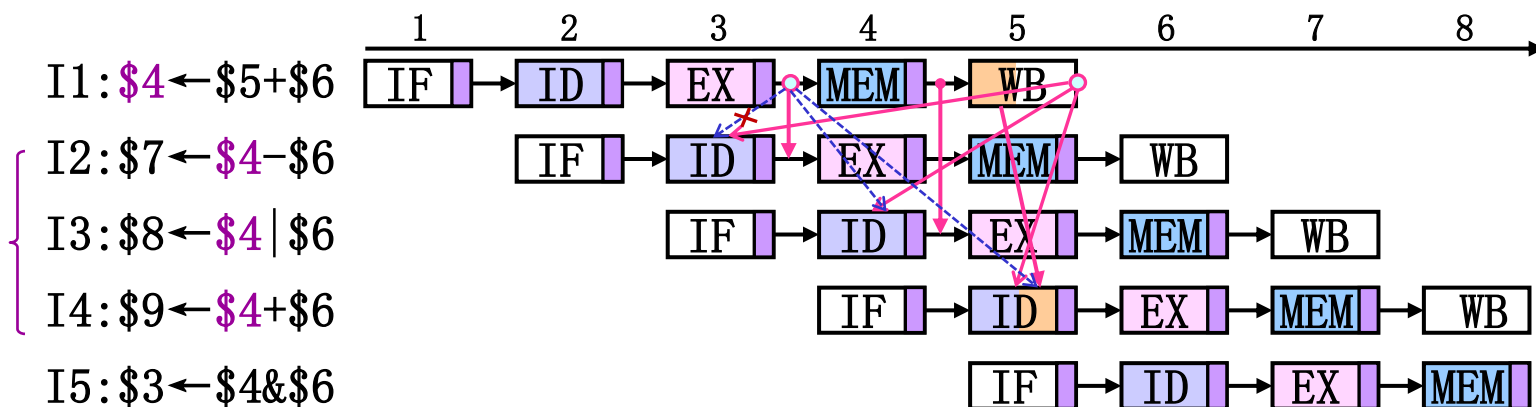
I1:	IF	ID	EX	MEM	WB				
I2:		IF	ID	ID	ID	ID	EX		
I3:			IF	IF	IF	IF	ID		

**\*转发法：**冲突指令可从数据产生段获取数据，来消除冒险

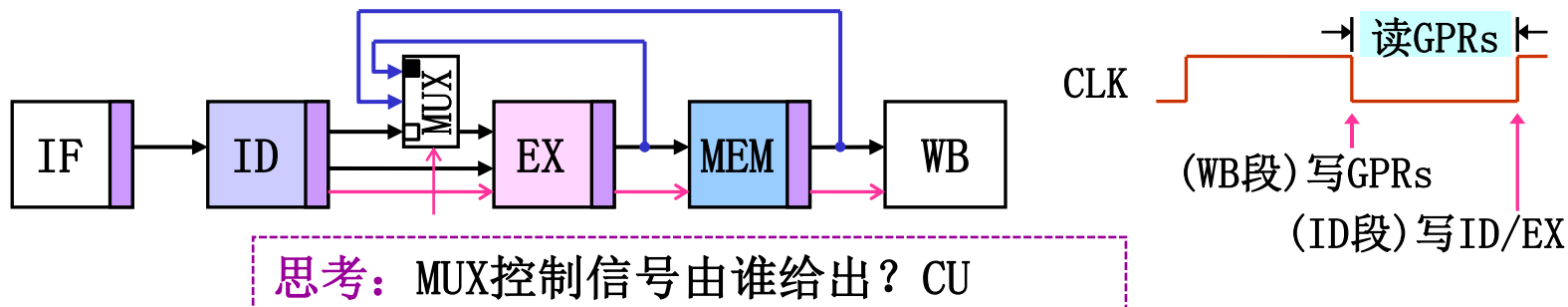
如EX段→┐ ┌→避免时间冲突→┐

获取时机优化—在使用时 (如EX段) 获取


←可减少冲突/简化实现



实现方法—增设转发线路 (到使用段)，同一拍中提前写GPRs



停顿拍数—可转发时=0拍，否则=阻塞法 ←产生段有EX及MEM

**例2：**续例1，RAW冒险采用转发法处理，写GPRs放在前半拍完成，  
 下列情况下，指令序列执行时间分别是多少？ P<sub>1</sub>时写 

(1)有EX→EX、MEM→EX转发线路 (2)仅有EX→EX转发线路

**解：** RAW冒险有：I2-I1、I3-I1、I4-I1，I5-I3

(1)I2-I1冒险停顿 0 拍； I3-I1冒险停顿 0 拍； (可转发)

I4-I1冒险停顿 0 拍； I5-I3冒险停顿 0 拍； (I4-I1同一拍)

执行时间 =  $[5t + (5-1)t] + (0+0+0+0)t = 9t$

(2)I2-I1冒险停顿 0 拍； I3-I1冒险停顿 1 拍； (ID读I1\_WB结果)

I4-I1冒险停顿 0 拍； I5-I3冒险停顿 1 拍； (同I3-I1)

执行时间 =  $[5t + (5-1)t] + (0+1+0+1)t = 11t$

**思考①：** 写GPRs放在后半拍时，小题(2)结果如何？

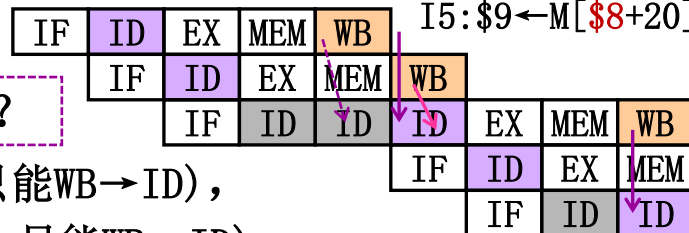
I3-I1、I5-I3各停顿2拍，I4-I1自动消除

**思考②：** I3为\$8←\$4 | \$7时，小题(2)结果如何？

I3取\$4停顿1拍(无MEM-EX转发、无EX-ID转发、只能WB→ID)，

取\$7再停顿1拍(无MEM-EX转发、无EX-ID转发、只能WB→ID)

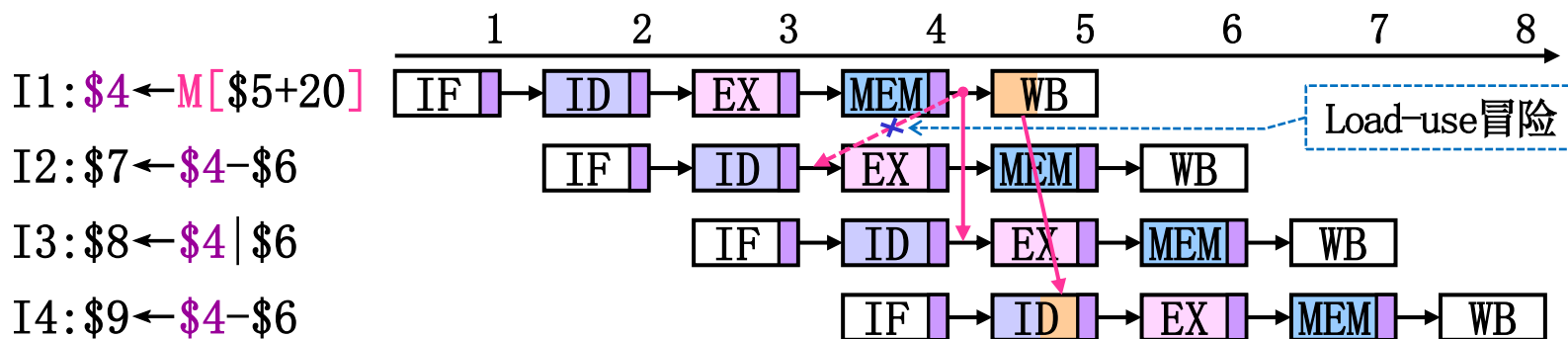
I1: \$4 ← \$5 + \$6  
 I2: \$7 ← \$4 - \$6  
 I3: \$8 ← \$4 | \$7  
 I4: M[\$4+20] ← \$6  
 I5: \$9 ← M[\$8+20]



思考①：  $T = [5t + (5-1)t] + (0+2+0+2)t = 13t$ ，思考②：  $T = [5t + (5-1)t] + (0+2+0+1)t = 12t$

# load-use冒险—lw指令引起、无法用转发法处理的RAW冒险

(仅紧邻指令[产生比使用迟1拍])



**处理方法：**阻塞法(插入气泡)，或软件方法(插入nop指令)

**练习：**MIPS流水线中，写GPRs放在后半拍实现，欲执行下列代码：

```

I1: add $4, $5, $6      ; $4 ← $5 + $6
I2: sub $7, $4, $6      ; $7 ← $4 - $6
I3: or  $8, $4, $6      ; $8 ← $4 | $6
I4: sw  $6, 20($4)      ; M[$4 + 20] ← $6
I5: lw  $6, 20($8)      ; $6 ← M[$8 + 20]
    
```

- (1) 哪些指令之间存在RAW冒险？
- (2) RAW冒险用阻塞法处理时，代码的执行时间？
- (3) RAW冒险用转发法处理(线路为EX→EX)时，代码的执行时间？
- (4) I4和I5对调后，小题(3)的结果？

(1) I2-I1、I3-I1、I4-I1, I5-I3 (2) I2-I1停3拍, I3-I1、I4-I1停0拍(随I2-I1消除), I5-I3停2拍  
 ;  $T = [5t + (5-1)t] + (3+2)t = 14t$ ; (3) I2-I1停0拍(EX-EX线路), I3-I1停2拍, I4-I1停0拍(随I3-I1消除), I5-I3停2拍;  
 $T = [5t + (5-1)t] + (2+2)t = 13t$ ; (4) I2-I1停0拍, I3-I1停2拍, I4-I3停0拍, I5-I4停3拍,  
 $T = [5t + (5-1)t] + (2+3)t = 14t$ 。



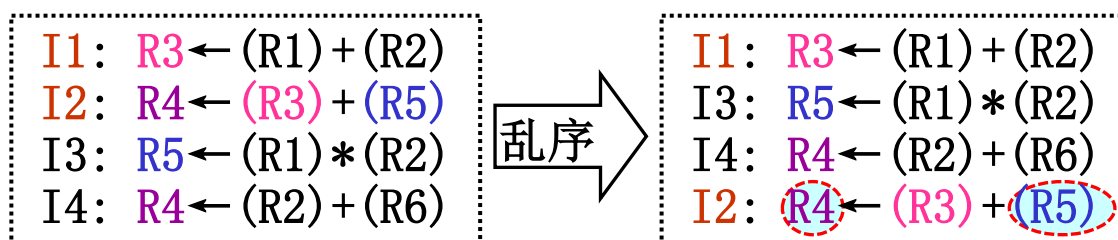
**\*乱序执行法：**冲突指令停顿，后续指令(无RAW冒险)可优先执行



**实现方法一** 增设指令窗口、采用动态调度方法  
(提供选择平台) (OPD就绪时流动)

**停顿拍数—0拍**

**新增冒险类型—读后写 (Write After Read, WAR) 冒险、  
写后写 (Write After Write, WAW) 冒险**



**处理方法：** 动态调度方法 (第4章讨论)

**作业1：** P91—5、6、11(1)(2)、12

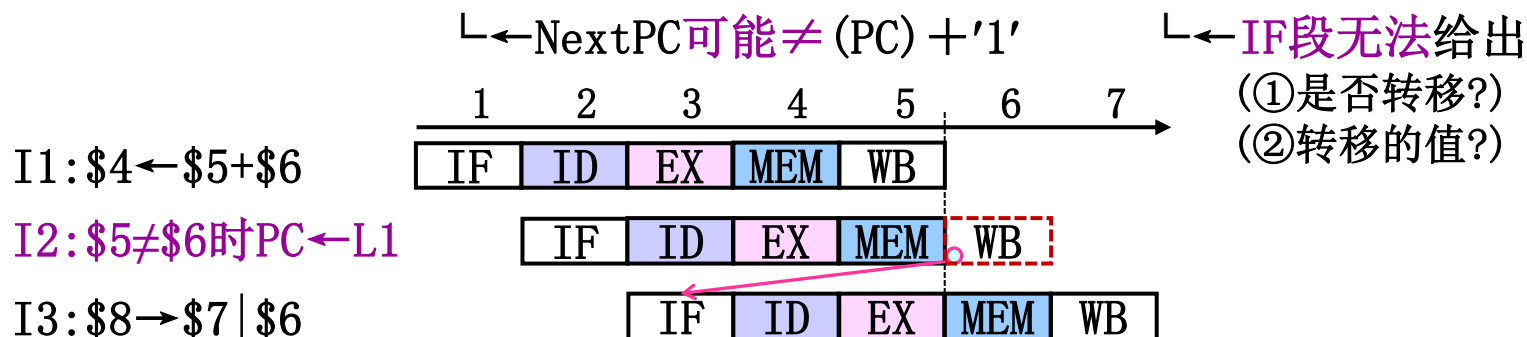
## 5、控制冒险处理

### 流水线功能例一

IF	$IR \leftarrow IM[(PC)], PC \text{ 及 } NPC \leftarrow (PC) + 4$
ID	$A \leftarrow R[rs], B \leftarrow R[rt], Imm \leftarrow ExtU(imme)$
EX	$C \leftarrow beq \cdot ((A) = (B)) ? 1 : 0, T \leftarrow (NPC) + Imm$
MEM	$C=1 \text{ 时 } PC \leftarrow T$

正常流水要求—IF段产生下条指令地址 (如PC+4)

**\*产生原因：** 因指令执行顺序改变，引起流水线停顿



**思考：** 上图中，WB段在第6拍的操作是什么？ 空操作 (Cmd为气泡)

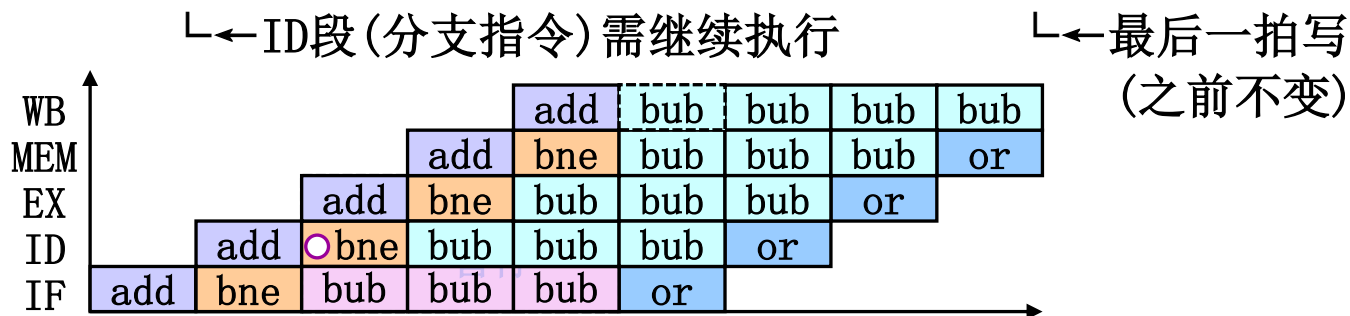
**\*处理方法：** 阻塞法、分支预测法、延迟分支法

$L \leftarrow$  又称冻结法、排空法

**\*阻塞法：**分支指令的后继指令**暂停，直到**冒险消除

### 冒险消除时要求—PC为分支结果，各段间REG均为气泡

## 暂停方法—IF段产生气泡 (nop指令)、PC为分支结果 ←锁步机制3



**思考①：**改为ID下拍产生气泡、IF暂停，可以吗？

可以，但很麻烦，且  
IF末拍需产生气泡

### 思考②：与数据冒险之锁步机制的差别？

气泡产生时机(非ID)/方式(非u0PCmd)、IF(无需暂停)

**停顿拍数**—从ID段到指令执行完的拍数( $=T_{\text{bne}}-1$ )

### 优化方法：①尽早判断是否转移

## ②尽早计算转移目标地址

} 需同时进行

**例3：** MIPS流水线中，有EX→EX的转发线路，分支指令在MEM段写PC。现有如下指令序列：

	addi \$4, \$5, 100	; I1: $\$4 \leftarrow \$5 + 100$
L1:	add \$8, \$6, \$7	; I2: $\$8 \leftarrow \$6 + \$7$
	sw \$8, 20(\$6)	; I3: $M[\$6 + 20] \leftarrow \$8$
	addi \$5, \$5, 1	; I4: $\$5 \leftarrow \$5 + 1$
	bne \$5, \$4, L1	; I5: $\$5 \neq \$4$ 时PC←L1
	addi \$9, \$9, 10	; I6: $\$9 \leftarrow \$9 + 10$

(1) 哪些指令之间存在RAW冒险？

(2) 控制冒险采用阻塞法处理时，指令序列的执行时间？

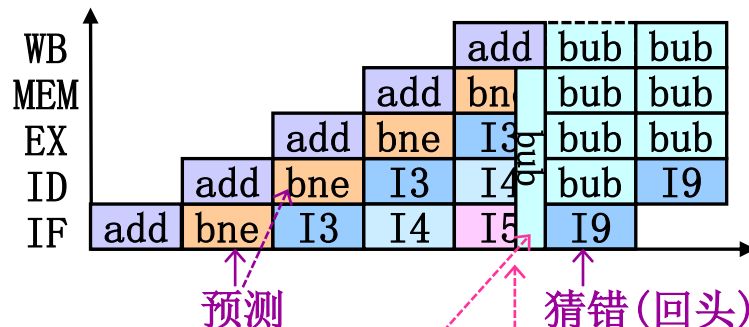
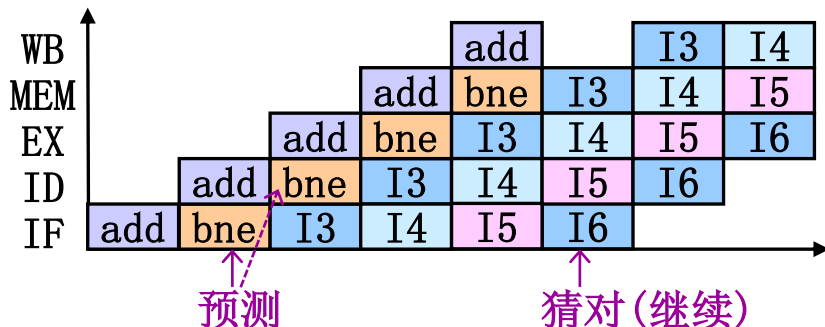
**解：** (1) RAW冒险有：I3-I2、I5-I4 (I5-I1有相关、无冒险)

(2) I3-I2冒险停顿 **0** 拍，I5-I4冒险停顿 **0** 拍；

控制冒险用阻塞法处理时，I5每次使流水线停 **3** 拍，

执行时间 =  $[5t + (402 - 1)t] + (0t + 0t + 3t) * 100 = 706t$

**\*预测法：** 预测转移方向 (转/不转)，并执行该方向的指令；  
猜对时继续执行后续指令，猜错时回头执行反方向指令



**预测策略一** (静态预测[盲猜]/动态预测[基于转移历史])

静态预测 { **预测不转移：** 无先决条件 (PC+1 已知)，可在 IF 段预测  
**预测转移：** 需先计算目标地址，最早在 ID 段预测

**动态预测：** 需保存转移历史&目标地址，可在 IF 段预测

**实现机制一** 猜对时不写PC，

猜错时重写PC、清空流水线 (产生气泡)

不占额外时间

**停顿拍数一** 猜对时  $\geq 0$  拍，猜错时 = 阻塞法 (到指令执行完为止)  
(ID 预测时 = 1 拍)  $\leftarrow$  无论是 IF 预测或 ID 预测  $\leftarrow$

**例3：** MIPS流水线有EX段→EX段转发线路， bne指令在ID段计算分支目标地址(专用部件)、在MEM段写PC。现有如下指令序列(同例2)：

L1: add \$8, \$6, \$7	; I2: \$8 ← \$6 + \$7, 假设\$4=100、\$5=0
sw \$8, 20(\$6)	; I3: M[\$6+20] ← \$8
addi \$5, \$5, 1	; I4: \$5 ← \$5 + 1
bne \$5, \$4, L1	; I5: \$5 ≠ \$4时PC ← L1
addi \$9, \$9, 10	; I6: \$9 ← \$9 + 10

(1)控制冒险用阻塞法处理时，代码执行时间？

(2)控制冒险在IF段预测(预测不转移)，代码执行时间？

(3)若在ID段预测(预测转移)，则预测的总停顿时间？

**解：** RAW冒险有I1-I2、I5-I4，冒险需停顿0拍(可转发)；

(1)每次控制冒险暂停 3 拍； $T = [5t + (400 + 1 - 1)t] + 3t * 100 = 705t$

(2)I5预测正确 1 次(停 0 拍/次)、预测错误 99 次(停 3 拍/次)；

$$T = [5t + (400 + 1 - 1)t] + 3t * 99 = 702t$$

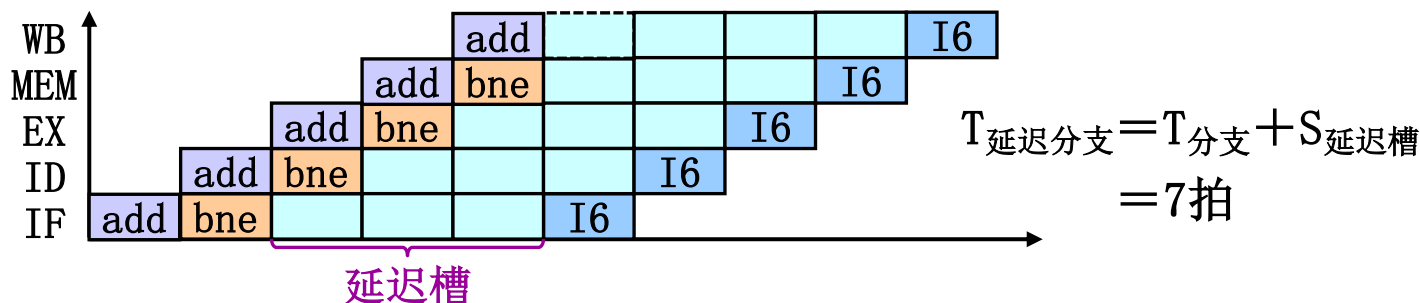
(3)I5预测正确时停 1 拍/次、错误时停 3 拍/次 (分支执行完即可回头)

$$T_{\text{停顿}} = 1t * 99 + 3t * 1 = 102t$$

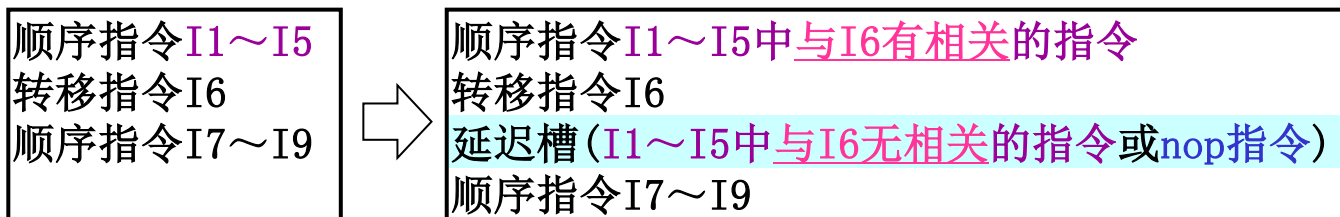
**\*延迟分支法：** 延迟槽中指令总是被执行，  
使延迟槽中尽量为有用指令

逻辑上延长了分支指令执行时间

**延迟槽**— 分支指令执行完前，可流入流水线的指令位置



**实现机制**— 软件实现 (编译时重排序指令序列)



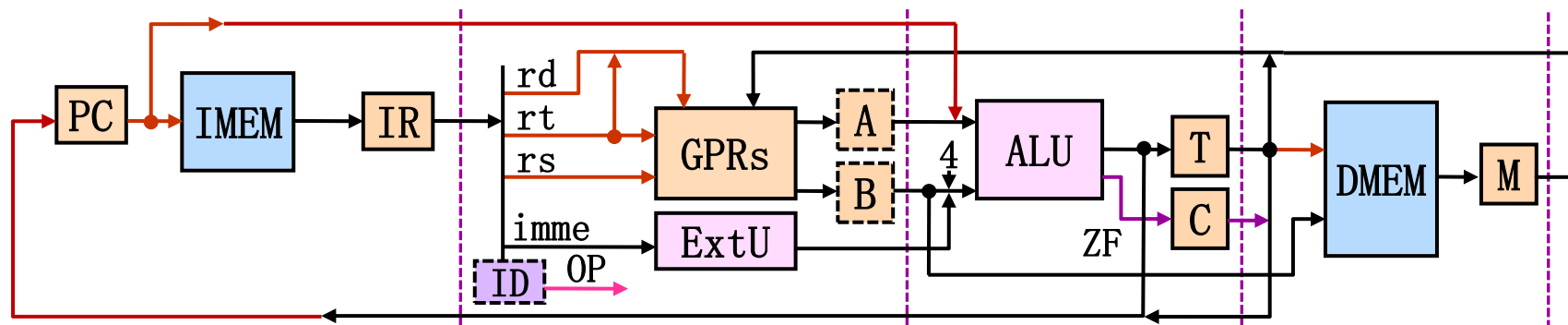
**停顿拍数**— 延迟槽中均为nop指令时，=阻塞法；  
延迟槽中含有用指令时，<阻塞法

**适用场合**— 延迟槽=1条指令时 (即  $T_{\text{分支}} = 2T_c$ ) ←性能损失(为nop)小  
← ≥2条时常用分支预测法 ←与延迟分支法不兼容

# 第3节 流水线的实现

※主要内容：流水线数据通路的实现，流水线控制器的实现

\*多周期数据通路回顾：（以MIPS为例）



指令执行过程—

	add: $rd \leftarrow (rs) + (rt)$ ori: $rt \leftarrow (rs) \mid \text{imme}$	lw: $rt \leftarrow M[(rs) + \text{imme}]$ sw: $M[(rs) + \text{imme}] \leftarrow (rt)$	if: $((rs) = (rt))$ $PC \leftarrow (PC) + 4 + \text{imme} \ll 2$
IF	$IR \leftarrow IM[(PC)],$		$PC \leftarrow (PC) + 4$
ID	$A \leftarrow R[rs], B \leftarrow R[rt],$		$T \leftarrow PC + \text{ExtU}$
EX	$T \leftarrow (A) \text{ op } (B) \text{ 或 ExtU}$	$T \leftarrow (A) + \text{ExtU}$	$C \leftarrow ((A) = (B)) ? 1 : 0$
MEM	$R[rd] \leftarrow (T)$	$M \leftarrow DM[T]$	$DM[T] \leftarrow (B)$ $PC \leftarrow T \text{ (C=1时)}$
WB		$R[rt] \leftarrow (M)$	

思考：上图 $PC \leftarrow (PC) + 4$ 等uOP使用的数据路径？结构冒险的表现？

ALU入端为PC及4，ALU出端直连PC；部件复用3次，ALU操作、写GPRs、写PC时间不唯一，T、B隔段使用<sup>32</sup>

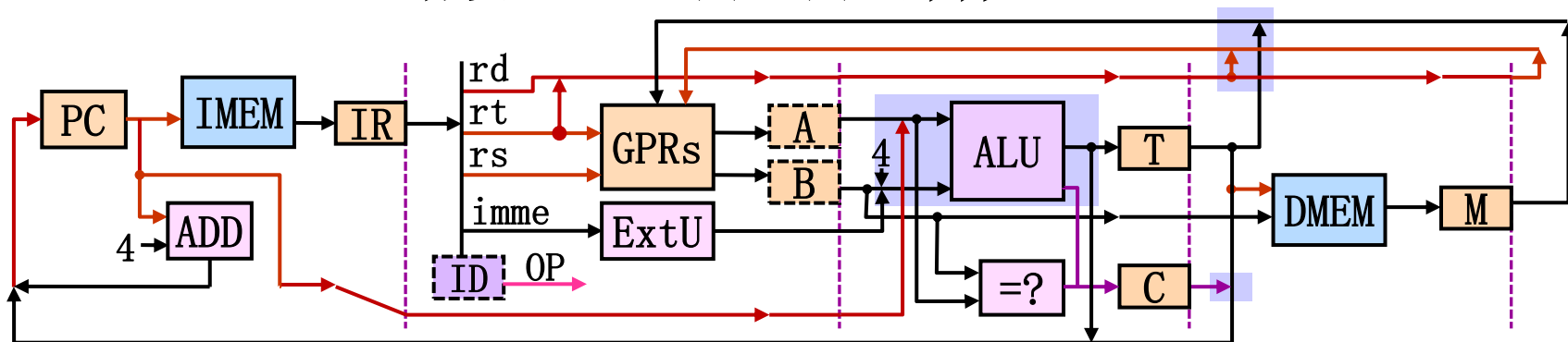


# 1、流水线数据通路的实现

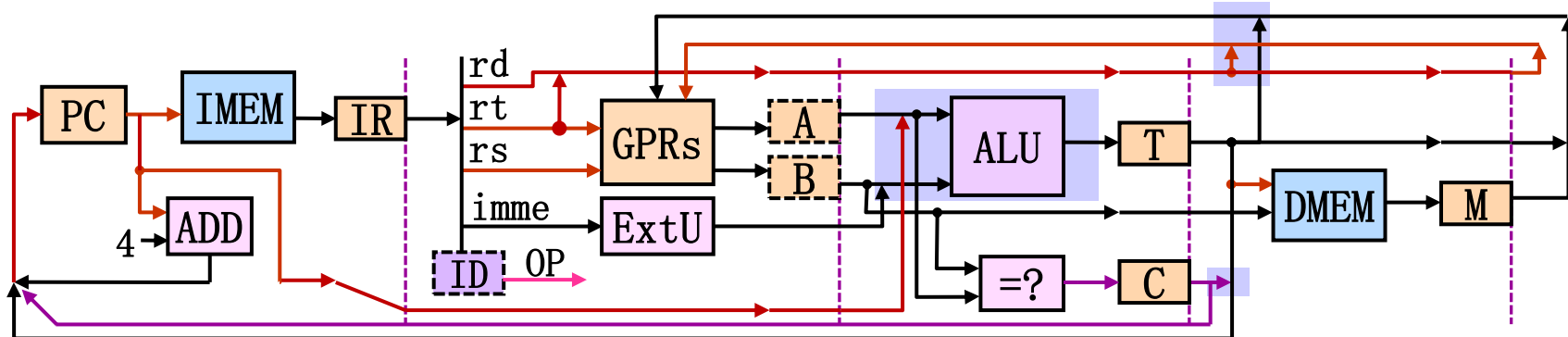
**\*步骤：** 构建基本通路(含结构冒险处理)， 增加数据/控制冒险处理通路

**\*基本通路组成及结构冒险处理：** (部件+互连组织)

部件复用处理—增设PC+4、 (A) = (B)? 部件



部件使用时间冲突处理— 写GPRs及ALU时间唯一、写PC在IF段(选数据)

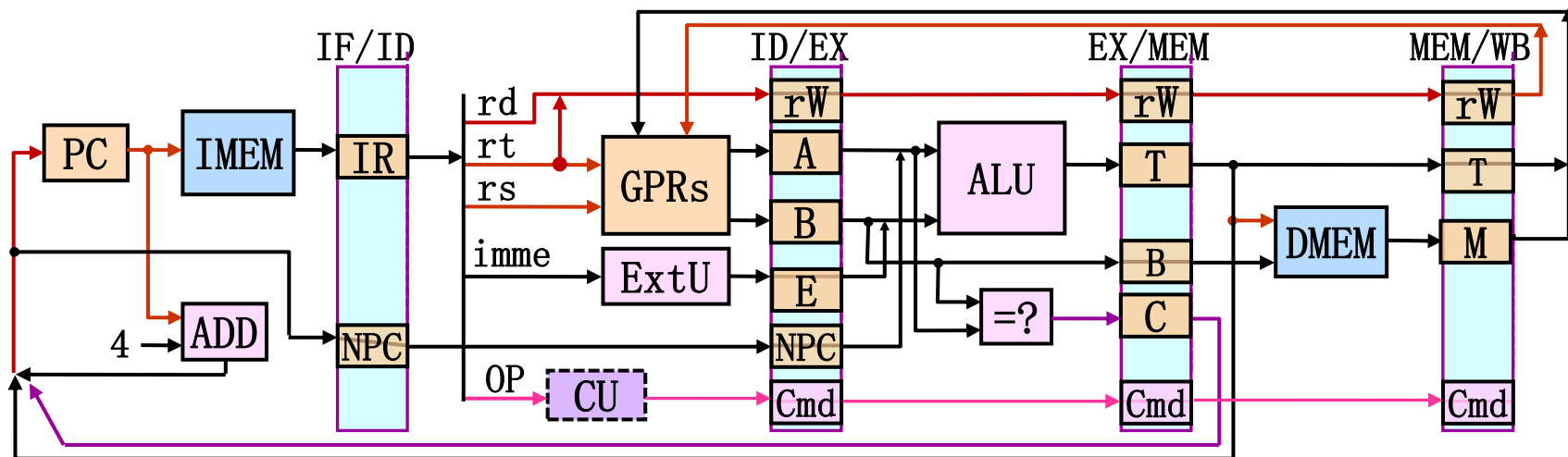


**思考：** 第1条指令IF段PC写的值？如何实现？ (PC) +4, C初值=0 (初始化时)

## 各段操作独立组织—设置段间寄存器(暂存数据/地址/命令)

└→边沿触发(拍结束时写)→┐

└←产生-使用间的所有段



## 各段功能组织—IF段源自PC, 其余段源自段间REG(传递信息未写)

段\指令	算逻运算 (add/ori)	存取 (lw/sw)	分支 (beq)
IF	IR ← IM[ (PC) ], <b>PC</b> 及 <b>NPC</b> ← <b>EX/MEM. C?</b> EX/MEM. T: (PC) + 4		
ID	A ← R[rs], B ← R[rt], E ← ExtU(imme), rW ← R型? rd:rt, Cmd ← CU		
EX	T ← (A) op (B) 或 (E)	T ← (A) + (E)	T ← ( <b>NPC</b> ) + (E) <b>C</b> ← ((A) = (B)) ? 1 : 0
MEM	<b>空闲</b>	M ← DM[T]	DM[T] ← (B) <b>空闲</b> (IF段使用数据)
WB	R[rW] ← (T)	R[rW] ← (M)	

## \*数据冒险处理:

在ID段

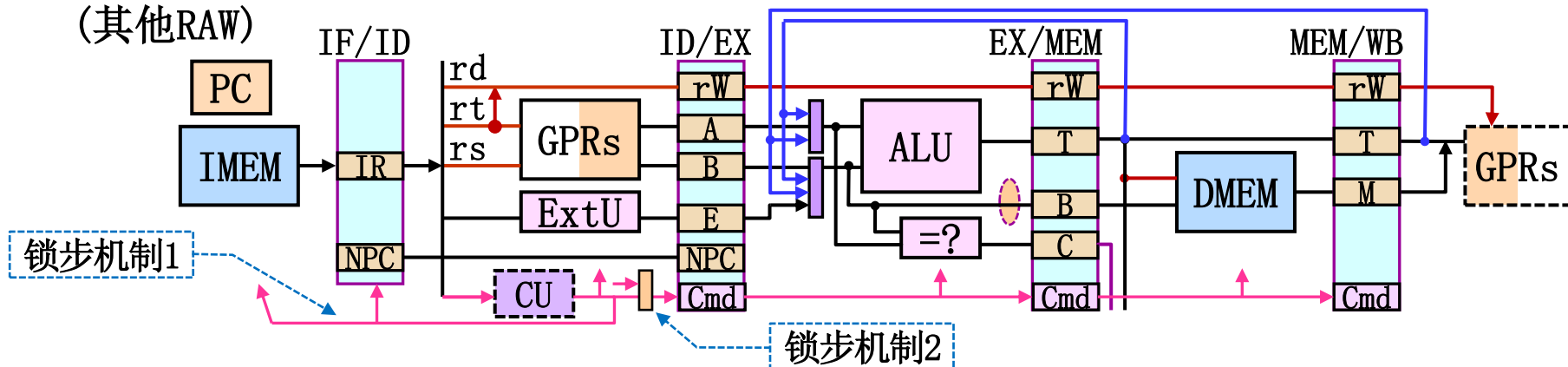
冲突检测—CU实现, IF/ID. IR. rs及rt=前驱指令目的OPD地址?

思考①: 图中表示上条、上上条指令目的OPD地址的信号线?

ID/EX. rW、  
EX/MEM. rW

转发法—增设EX段转发线路, 在前半拍写GPRs

(其他RAW)



思考②: GPRs在前半拍写如何实现?

CLK

读GPRs

写GPRs

写ID/EX

思考③: EX/MEM. B连接MUX(而非ID/EX. B)的好处?

可处理add-sw的RAW

阻塞法—CU实现, ID段产生气泡、IF段停顿, 停1拍(当前拍)

(load-use)

└→ 锁步机制1 (不写IF/ID及PC)

└→ 锁步机制2 (气泡写入ID/EX. Cmd)

思考④: ID段检测到冒险, 同一拍为何能使IF段停顿?

CU产生Cmd

写IF/ID

思考①: ID/EX. rW、EX/MEM. rW。思考②: GPRs的Clk接P1引脚。思考③: 可处理add-sw指令的转发。

思考④: Cmd产生延迟很小(由组合逻辑电路产生), CLK结束时才写IF/ID REG, 来得及。

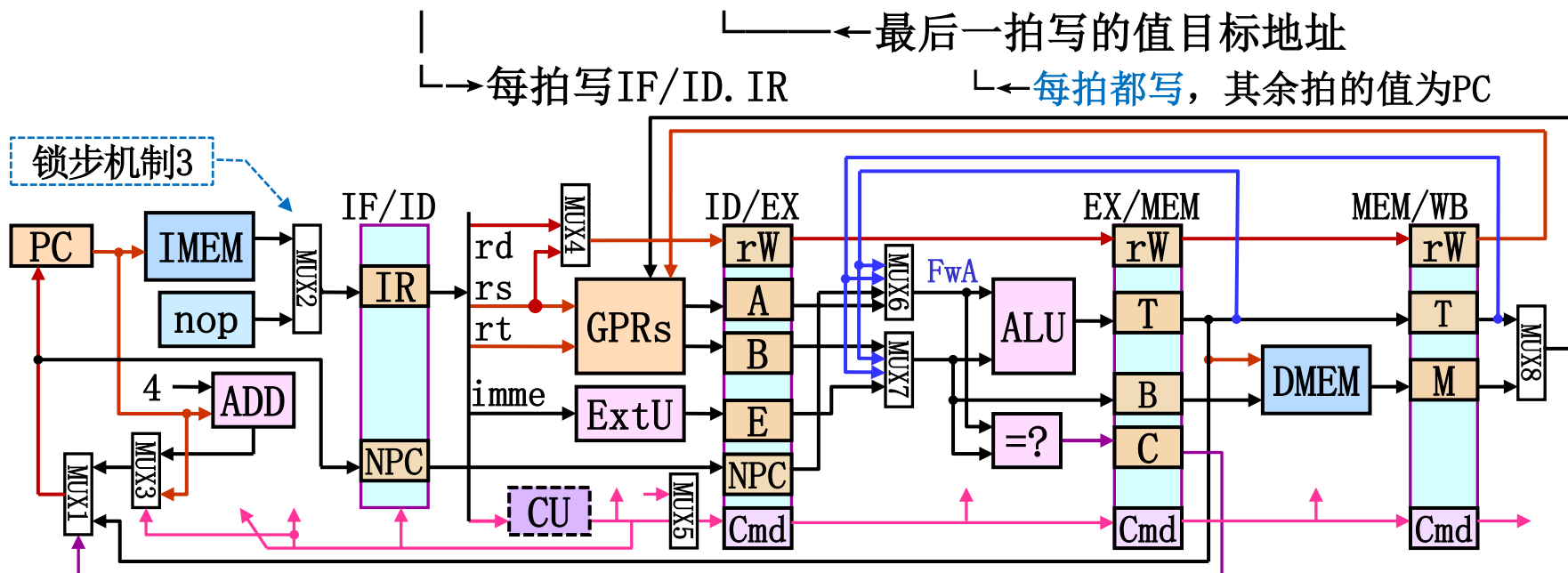
## \*控制冒险处理:

在ID、EX、MEM段

(CU. Brn | EX/MEM. Brn | MEM/WB. Brn) 产生气泡, Brn为beq的uOPCmd

冲突检测—CU实现, 当前指令=(beq | bne | j)? 思考①: 逻辑表达式?

阻塞法—IF段产生气泡、PC为分支结果, 停3拍(当前起) ←锁步机制3

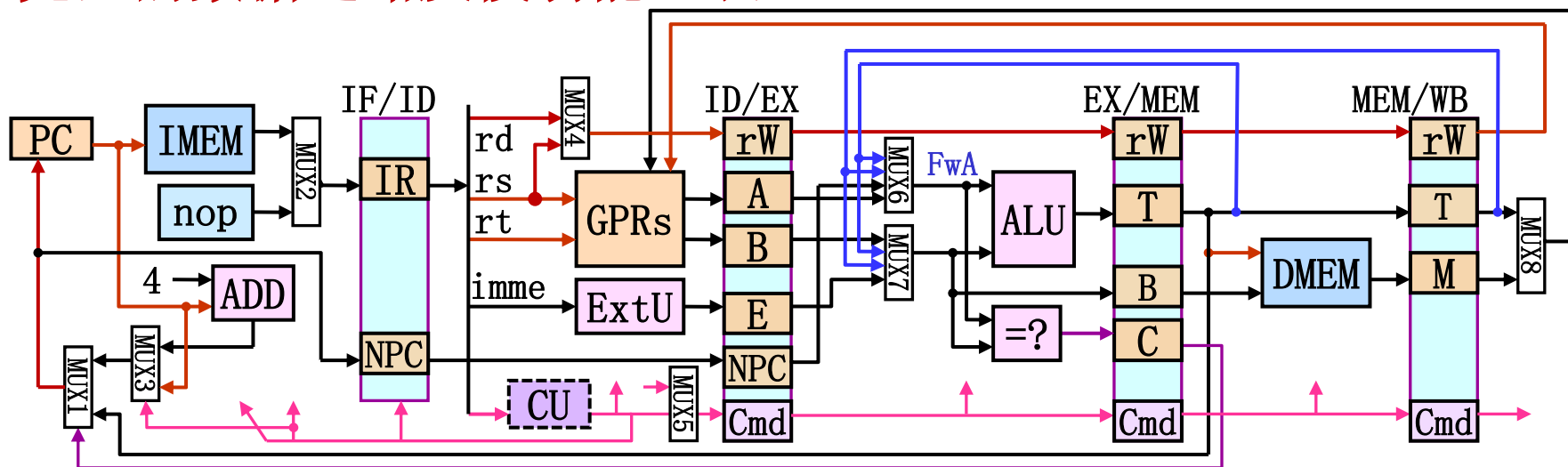


思考: 最后一拍写如何实现? 最后一拍的表示? EX/MEM. Cmd=beq | bne时

优化: EX段写PC(停2拍), 需保证拍时长(EX段+MUX1)、拍结束时写;  
ID段写PC(停1拍), 需前移比较器、增设加法器、写PC同上

预测法—(第4章讨论)

## \*完整的数据通路及段功能组织：（IF段源自PC，其余段源自段间REG）



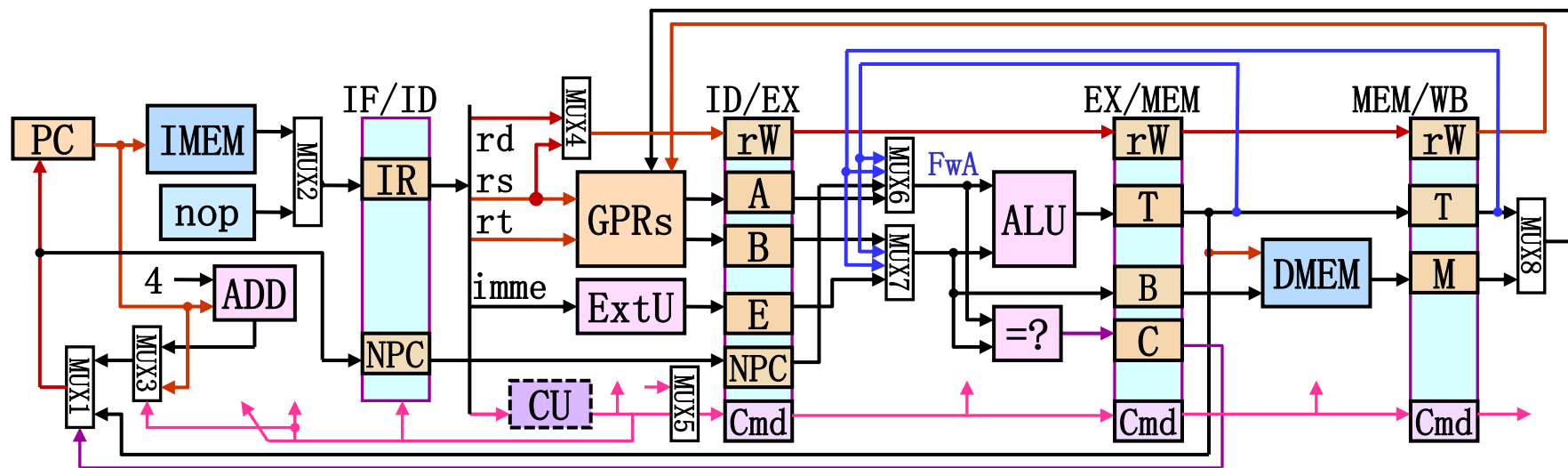
控制冒险

PC保持不变

Load-use冒险

IF	PC及.NPC $\leftarrow$ EX/MEM. C? EX/MEM. T : (CHazd[非末拍]? PC:PC+4); . IR $\leftarrow$ CHazd? nop:IM[PC]; . Cmd (IF/ID及PC) $\leftarrow$ DHazd? 0:1
ID	. A $\leftarrow$ R[rs], . B $\leftarrow$ R[rt], . E $\leftarrow$ ExtU(imme); . NPC $\leftarrow$ IF/ID. NPC, . rW $\leftarrow$ rt或rd; . Cmd $\leftarrow$ DHazd? nop:IR. op(含转发)
EX	. T $\leftarrow$ (FwA) op (FwB), . B $\leftarrow$ FwB, . C $\leftarrow$ (ID/EX. Cmd=beq · FwA=FwB)? 1:0; . rW $\leftarrow$ ID/EX. rW; . Cmd $\leftarrow$ ID/EX. Cmd子集
MEM	. M $\leftarrow$ DM[EX/MEM. T] 或 M[EX/MEM. T] $\leftarrow$ EX/MEM. B, . T $\leftarrow$ EX/MEM. T; . rW $\leftarrow$ EX/MEM. rW; . Cmd $\leftarrow$ EX/MEM. Cmd子集
WB	R[rW] $\leftarrow$ MEM/WB. T 或 MEM/WB. M

**例1:** MIPS流水线中, GPRs在前半拍写, RAW冒险用转发法处理, 控制冒险用阻塞法处理, 写出下列代码执行时流水线各拍的状态。



I1: \$1 ← \$0 + 40  
 I2: \$2 ← M[\$0 + 4]  
 I3: \$3 ← \$1 - \$2  
 I4: \$4 ← \$1 + \$3  
 I5: \$5 = \$5 时 PC ← I2

### 流水线操作说明:

- (1) **操作的数据**— IF段源自PC, 其余段源自段间REG
- (2) **转发的数据**— 源自EX/MEM或MEM/WB, 到达EX段
- (3) **操作的控制**— IF&ID段基于CU, 其余基于段间REG. Cmd
- (4) **气泡的产生**— CU控制, 写入IF/ID. IR或ID/EX. Cmd
- (5) **IF段的停顿**— CU控制, 不写IF/ID及PC

R型	op	rs	rt	rd	shm	func
I型	op	rs	rt	imme/disp		
J型	op	addr				

段	段间REG逻辑		初始	1	2	3	4	5	6
IF	=EX/MEM. C? EX/MEM. T: PC+4或PC(CHazd时)	PC	0	4	8	12	12	16	20
		NPC	*	4	8	12	12	16	20
	=CHazd? nop:IM[PC]	IR	nop	(I1)	(I2)	(I3)	(I3)	(I4)	(I5)
ID	=R[IR. rs]	A	*	*	0	?	0	0	22⑥
	=R[IR. rt]	B	*	*	*	*	*	?	40⑦
	=ExtU(IR. imme)	E	*	*	40	4	*	*	*
	=IR. op? IR. rt:IR. rd	rW	*	*	1	2	3	3	4
	=DHazd? nop:IR. op(含转发)	Cmd	nop	nop	ADDI	LW转②	nop④	SUB转②	ADD
	=IF/ID. NPC	NPC	*	*	4	8	12	12	16
EX	=FwA op FwB	T	*	*	*	40	44③	*	-22⑤
	=beq · (FwA=FwB)? 1:0	C	0	0	0	0	0	0	0
	=FwB	B	*	*	*	*	*	*	22
	=ID/EX. rW	rW	*	*	*	1	2	3	3
	=ID/EX. Cmd子集	Cmd	nop	nop	nop	ADDI	LW	nop	SUB
MEM	=DM[EX/MEM. T]①	M	*	*	*	*	*	22	*
	=EX/MEM. T	T	*	*	*	*	40	44	*
	=EX/MEM. rW	rW	*	*	*	*	1	2	3
	=EX/MEM. Cmd子集	Cmd	nop	nop	nop	nop	ADDI	LW	nop
WB	R[E/M. rW] ← MEM/WB. T或M			nop	nop	nop	nop	\$1=40	\$2=22

I1:  $\$1 \leftarrow \$0 + 40$     注: ①写MEM功能未列出,    \*-表示当前指令未使用,    ?-表示数据尚未就绪  
 I2:  $\$2 \leftarrow M[\$1 + 4]$     ②其他RAW(Cmd中含转发路径)    ③转发(EX/MEM. T+ID/EX. E=40+4)  
 I3:  $\$3 \leftarrow \$0 - \$2$     ④load-use(插入气泡/IF暂停)    ⑤转发(MEM/WB. T+ID/EX. E=0-22)  
 I4:  $\$4 \leftarrow \$2 + \$1$     ⑥同一拍写后可读    ⑦相关≠RAW(\$1已就绪)  
 I5:  $\$5 = \$5$ 时PC ← I2

段	段间REG逻辑		重复6	7	8	9	10	11
IF	=EX/MEM. C? EX/MEM. T: PC+4或PC(CHazd时)	PC	20	20①	20①	4③	8	
		NPC	20	20	20	4	8	
	=CHazd? nop:IM[PC]	IR	(I5)	nop①	nop①	nop①	(I2)	
ID	=R[IR. rs]	A	22⑥	(\$5)	*	*	*	40
	=R[IR. rt]	B	40⑦	(\$5)	*	*	*	*
	=ExtU(IR. imme)	E	*	-16	*	*	*	4
	=IR. op? IR. rt:IR. rd	rW	4	*	*	*	*	2
	=DHazd? nop:IR. op(含转发)	Cmd	ADD	BEQ	nop	nop	nop	LW
	=IF/ID. NPC	NPC	16	20	20	20	20	8
EX	=FwA op FwB	T	-22⑤	62	4②	*	*	*
	=beq · (FwA=FwB)? 1:0	C	0	0	1	0	0	0
	=FwB	B	22	40	*	*	*	*
	=ID/EX. rW	rW	3	4	*	*	*	*
	=ID/EX. Cmd子集	Cmd	SUB	ADD	BEQ	nop	nop	nop
MEM	=DM[EX/MEM. T]①	M	*	*	*	*	*	*
	=EX/MEM. T	T	*	-22	62	*	*	*
	=EX/MEM. rW	rW	*	3	4	*	*	*
	=EX/MEM. Cmd子集	Cmd	nop	SUB	ADD	BEQ③	nop	nop
WB	R[E/M. rW] ← MEM/WB. T或M		\$2=22	nop	\$3=-22	\$4=62	nop	nop

I1: \$1 ← \$0+40

I2: \$2 ← M[\$1+4]

I3: \$3 ← \$0-\$2

I4: \$4 ← \$2+\$1

I5: \$5 = \$5时PC ← I2

注: ①有控制冒险时, IF段产生气泡、PC保持不变(非末拍)

②ID/EX. NPC+ID/EX. E=20+FFFC

③控制冒险末拍时(进入MEM段), PC为分支结果



## \*流水线数据通路实现小结:

功能段的划分— 5个段, 分支/sw指令仅用4个段(WB段为nop)

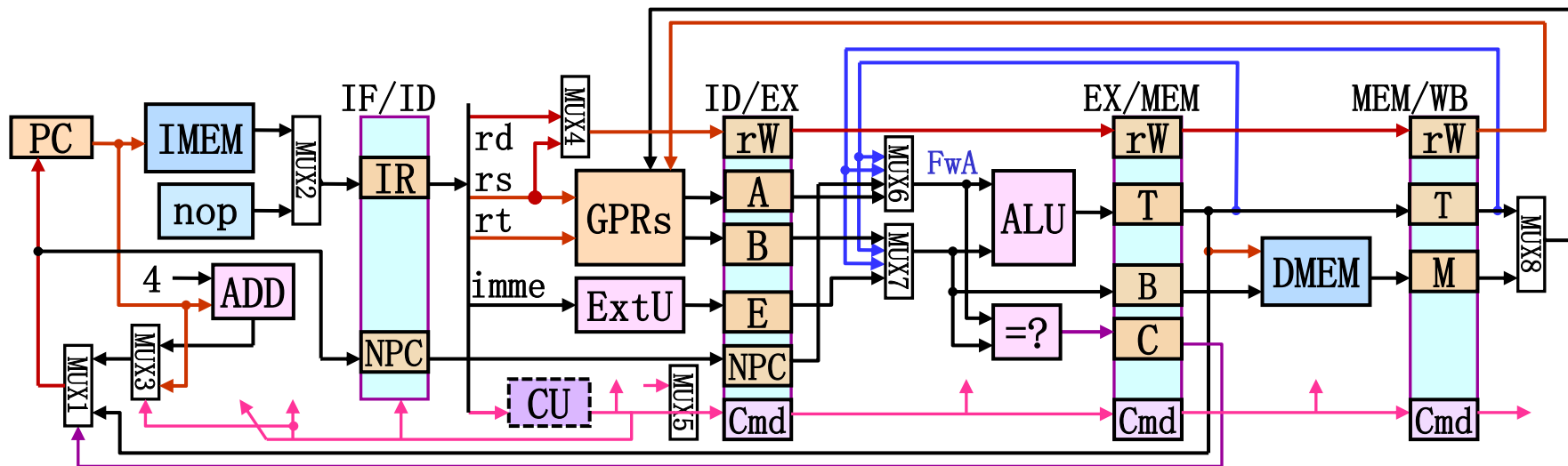
结构冒险处理— 部件无复用、使用时间唯一, 设置段间REG(D/A/C)

RAW冒险处理— 转发法(增设转发通路&前半拍写GPRs) +

阻塞法(IF段暂停&ID产生气泡, 1拍)

←锁步机制1&2

控制冒险处理— 阻塞法(IF段产生气泡&PC为分支结果, 3拍) ←锁步机制3



## 操作实现的组织—

各段的操作：IF段基于PC及EX/MEM、其余段基于段间REG

操作的控制：IF及ID段源自CU，其余段源自段间REG

## 阻塞功能汇总—

锁步机制1：IF段暂停(不写IF/ID及PC)，直至RAW冒险消除

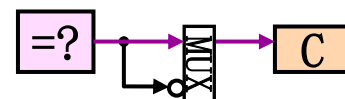
锁步机制2：ID段产生气泡(写ID/EX. Cmd)，直至RAW冒险消除

锁步机制3：IF段产生气泡(写IF/ID. IR)、PC为分支结果，  
直至控制冒险消除

## μOPCmd组织—

	指令功能μOPCmd	冒险处理μOPCmd
IF		PCsrc, PCWr, IRsrc, IF/IDctr
ID	rWsrc, ExtUOp	无(气泡uOPCmd由CU产生)
EX	ALUAsrc, ALUBsrc, ALUctr	无
MEM	MemRd, MemWr	无
WB	RegDsrc, RegWr	无

思考：增加bne指令时，数据通路的变化？需增加μOPCmd吗？



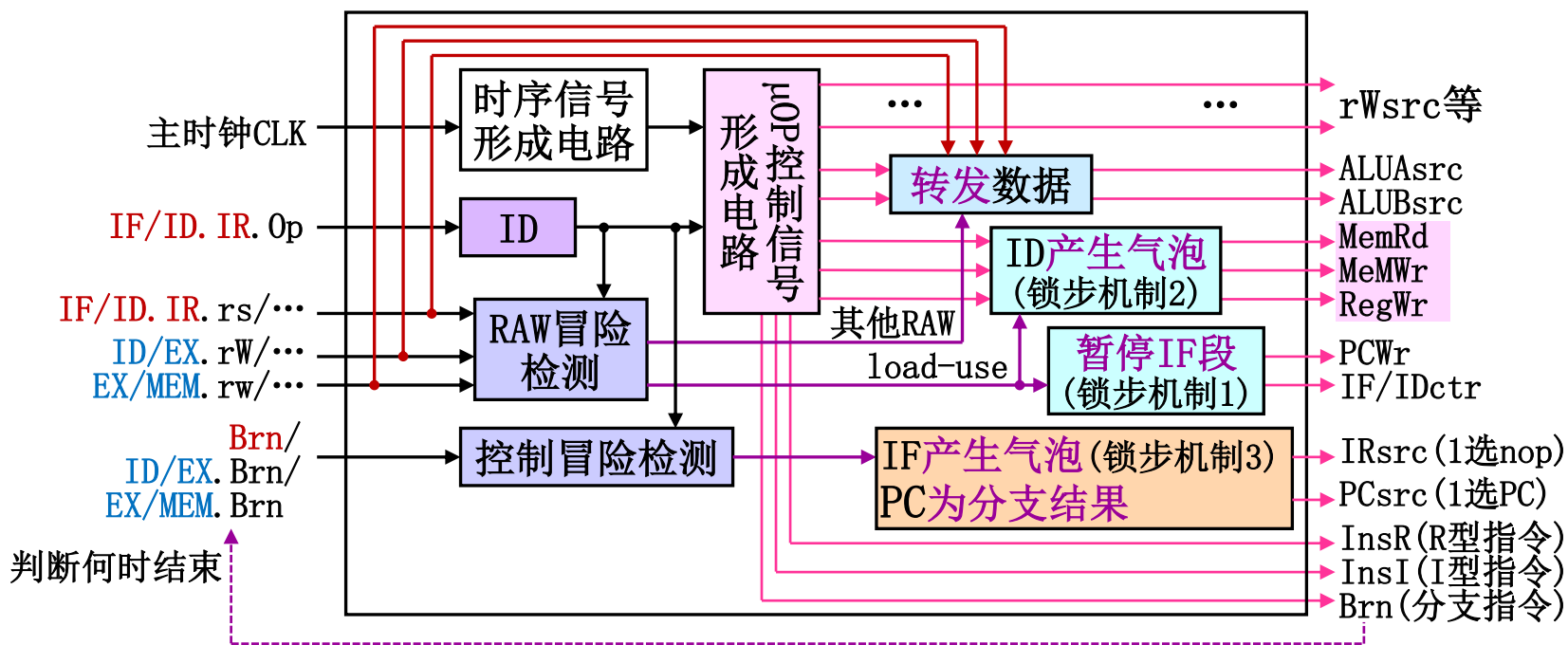
思考-EX/MEM. C前增加2路MUX，需增加1个指令类型信号。

## 2、流水线控制器的实现

**\*CU组成：** ID+时序电路+uOP控制电路+冒险检测/控制电路  
(一级时序)      (产生 $\mu\text{OPCmd}$ )      (产生冒险控制信号)

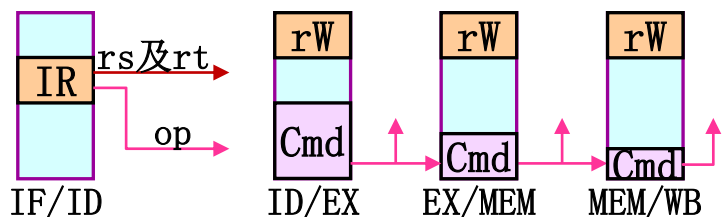
冒险检测—RAW冒险(load-use/其他)；控制冒险

冒险控制—转发, IF段暂停(锁步机制1)、ID段产生气泡(锁步机制2)；  
IF段产生气泡(锁步机制3)、PC为分支结果(未拍前不变)



**思考：** InsR、InsI、Brn的逻辑函数？  $\text{InsR} = \text{add}$ ,  $\text{InsI} = \text{ori} | \text{lw} | \text{sw}$ ,  $\text{Brn} = \text{beq}$

## \*RAW冒险检测/控制:



R型	op	rs	rt	rd	shm	func
I型	op	rs	rt	imme/dis		
J型	op	addr				

add	$rd \leftarrow (rs) + (rt)$
ori	$rt \leftarrow (rs) \mid \text{ZExt}(\text{immed})$
lw	$rt \leftarrow M[(rs) + \text{SExt}(\text{immed})]$
sw	$M[(rs) + \text{SExt}(\text{immed})] \leftarrow (rt)$
beq	if $((rs) = (rt))$ $PC \leftarrow (PC) + 4 + \text{SExt}(\text{immed}) \ll 2$

思考①: 表示上条、上上条指令目的OPD地址的信号线? 表示上条指令为lw、I型、分支指令的函数? 表示上条指令需写GPRs的函数?

思考②: 当前指令rt为源OPD地址的函数?  $\text{rtHav} = \text{InsR} \mid (\text{InsI} \cdot \text{MemWr})$

思考③: 判断当前指令rs与上条、上上条指令目的OPD地址相同的函数?

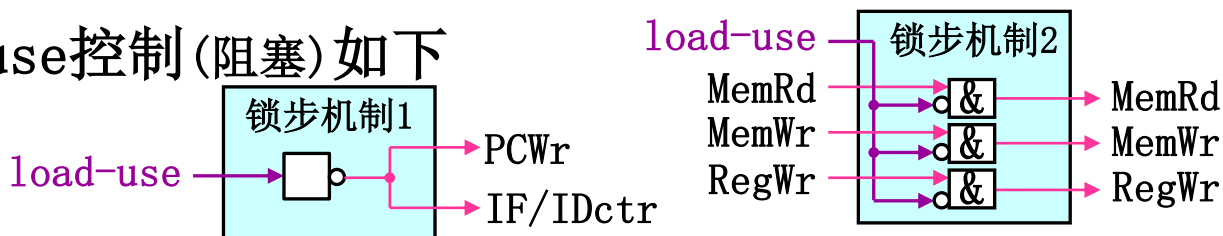
$\text{rsSamF1} = (\text{IF/ID. IR. rs} = \text{ID/EX. rW})$ ,  $\text{rsSamF2} = (\text{IF/ID. IR. rs} = \text{EX/MEM. rW})$

冒险检测—load-use =  $\text{ID/EX. MemRd} \cdot (\text{rsSamF1} \mid \text{rtHav} \cdot \text{rtSamF1})$

其他RAW =  $\sim \text{ID/EX. MemRd} \cdot \text{ID/EX. RegWr} \cdot (\text{rsSamF1} \mid \text{rtHav} \cdot \text{rtSamF1})$   
 $\mid \text{EX/MEM. RegWr} \cdot (\text{rsSamF2} \mid \text{rtHav} \cdot \text{rtSamF2})$

冒险控制—其他RAW控制(转发)ALUAsrc及ALUBsrc,

load-use控制(阻塞)如下



思考①: ID/EX. rW, ID/EX. MemRd、ID/EX. InsI、ID/EXBrn, ID/EX. RegWr.

## \*控制冒险检测/控制:

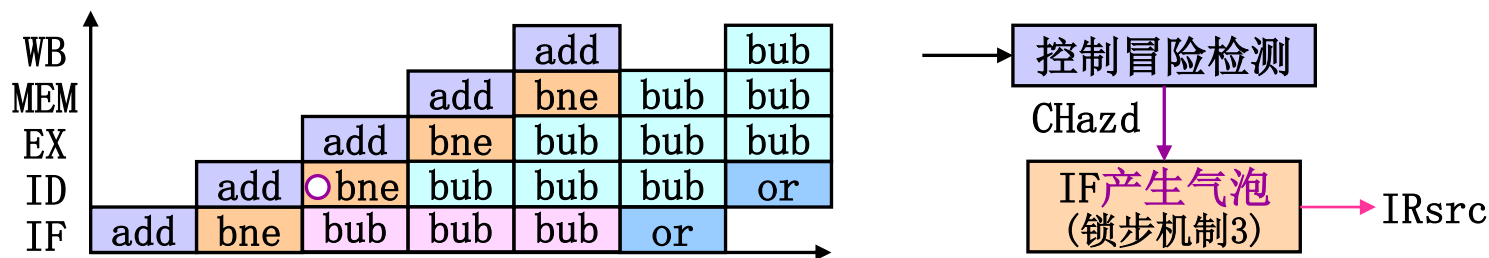
思考①: 如何表示分支指令执行到ID段、EX段?

Brn=1、ID/EX. Brn=1

思考②: 下列检测/处理方案哪个更容易实现?

a) ID段检测一次 (CHazd有效1个 $T_c$ ), IRsrc输出有效3个 $T_c$ ;

b) ID段一直检测 (CHazd有效3个 $T_c$ ), IRsrc输出依赖于CHazd



冒险检测—冒险消除前一直检测 (冒险控制为组合逻辑)

$$CHazd = Brn \mid ID/EX. Brn \mid EX/MEM. Brn$$

冒险控制— $IRsrc = CHazd$

←假设nop指令字连接MUX入端1

$$PCsrc = CHazd \cdot \sim EX/MEM. Brn \quad \leftarrow \text{未拍前PC不变 (连接MUX入端1)}$$

## \*控制信号形成逻辑汇总： --均ID段产生，EX、MEM、WB依赖于段间REG

使用段	控制信号	产生逻辑	冒险相关	功能
IF	PCWr	$\sim 1uRAW$	load-use	写PC
	IF/IDctr	$\sim 1uRAW$	load-use	写IF/ID
	IRsrc	CHazd	控制	IF段产生气泡
	PCsrc	CHazd · $\sim EX/MEM.Br_n$	控制	PC末拍前不变
ID	rWsrc	IR. op=add	/	目的OPD地址选择
	ExtUop	IR. op $\neq$ ori	/	ExtU控制
	InsR	IR. op=add	/	指令为R型
	InsI	IR. op=ori   lw   sw	/	指令为I型
	Brn	IR. op=beq	/	指令为分支型
EX	ALUAsrc	类似fwRAW之rs	其他RAW	ALU-A端断则
	ALUBsrc	类似fwRAW之rt		ALU-B段选择
	ALUctr	基于IR. op产生	/	ALU控制
MEM	MemRd	(IR. op=lw) · $\sim 1uRAW$	load-use	读DMEM
	MemWr	(IR. op=sw) · $\sim 1uRAW$	load-use	写DMEM
WB	RegWr	(IR. op=add   ori   lw) · $\sim 1uRAW$	load-use	写GPRs
	RegDsrc	IR. op=add	/	选择所写数据
冒险检测逻辑	1uRAW	同PPT3-P45之数据冒险检测		load-use检测结果
	fwRAW			其他RAW检测结果
	CHazd	Brn   ID/EX. Brn   EX/MEM. Brn		控制冒险检测结果

作业2：PPT—4，5

## 流水线实现方法研讨

**要求：** 已存在单周期数据通路(SDP)、多周期数据通路(MDP)，  
要实现相应的流水线数据通路(PDP)、控制器(PCU)

**讨论内容：**

- (1)SDP改为MDP/PDP时，时钟周期如何确定？
- (2)MDP改为PDP时，包含哪些步骤？
- (3)PDP中的段(如MEM段)时延不等长时，如何处理？
- (4)流水线需一直流动，阻塞法的核心思想是什么？
- (5)PCU只在ID段产生uOPCmd，如何保证指令的重叠执行？
- (6)PCU在哪个段检测冒险？如何获得前几条指令的信息？
- (7)PCU如何实现转发控制？如何实现阻塞控制？

### 第三章课后复习思考题

- 1、流水线的工作原理、组成基本要求？有哪些分类？
- 2、流水线的性能指标？提高性能指标的方法是什么？
- 3、流水线有哪几种冒险？产生原因？有哪些解决方法？
- 4、图3.32的MIPS流水线中，“符号位扩展”的功能有哪些？
- 5、基于图3.32，增加j指令的数据通路，写出IF段MUX的控制逻辑。