

## 本节主题



# 控制冒险的处理

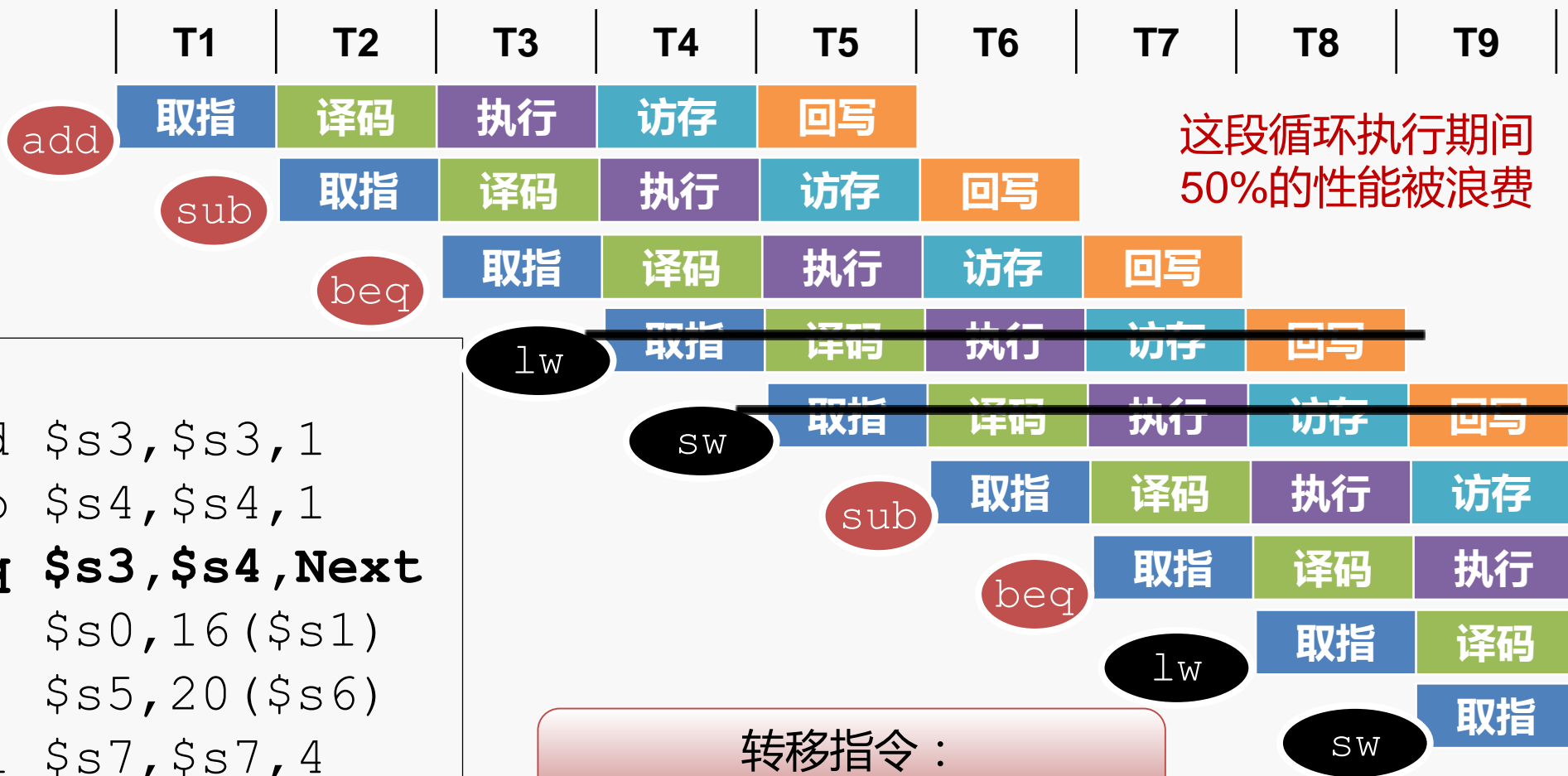
流水线重点内容，曾考过大题和选择。  
熟练掌握并理解流水线的三种冒险及解决方案

北京大学·慕课  
计算机组成  
制作人：陆俊林



# 转移指令对流水线的影响

流水线的最终性能目标：最大的指令吞吐率



```
...  
add $s3, $s3, 1  
Next: sub $s4, $s4, 1  
beq $s3, $s4, Next  
lw $s0, 16($s1)  
sw $s5, 20($s6)  
ori $s7, $s7, 4  
...
```

转移指令：  
改变指令流向，破坏流水模式

# 转移指令对性能的影响

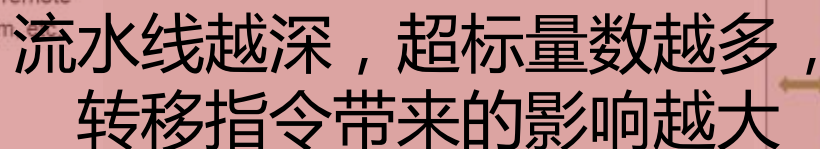
$$\begin{aligned} &\text{Pipeline stall cycles from branches} \\ &= \text{Branch Frequency} \times \text{Branch Penalty} \end{aligned}$$

## 🕒 转移指令所占比例 ( Branch Frequency )

- 每隔4到7条指令就会有一条转移指令
- 转移指令所占比例大约为15%~25%

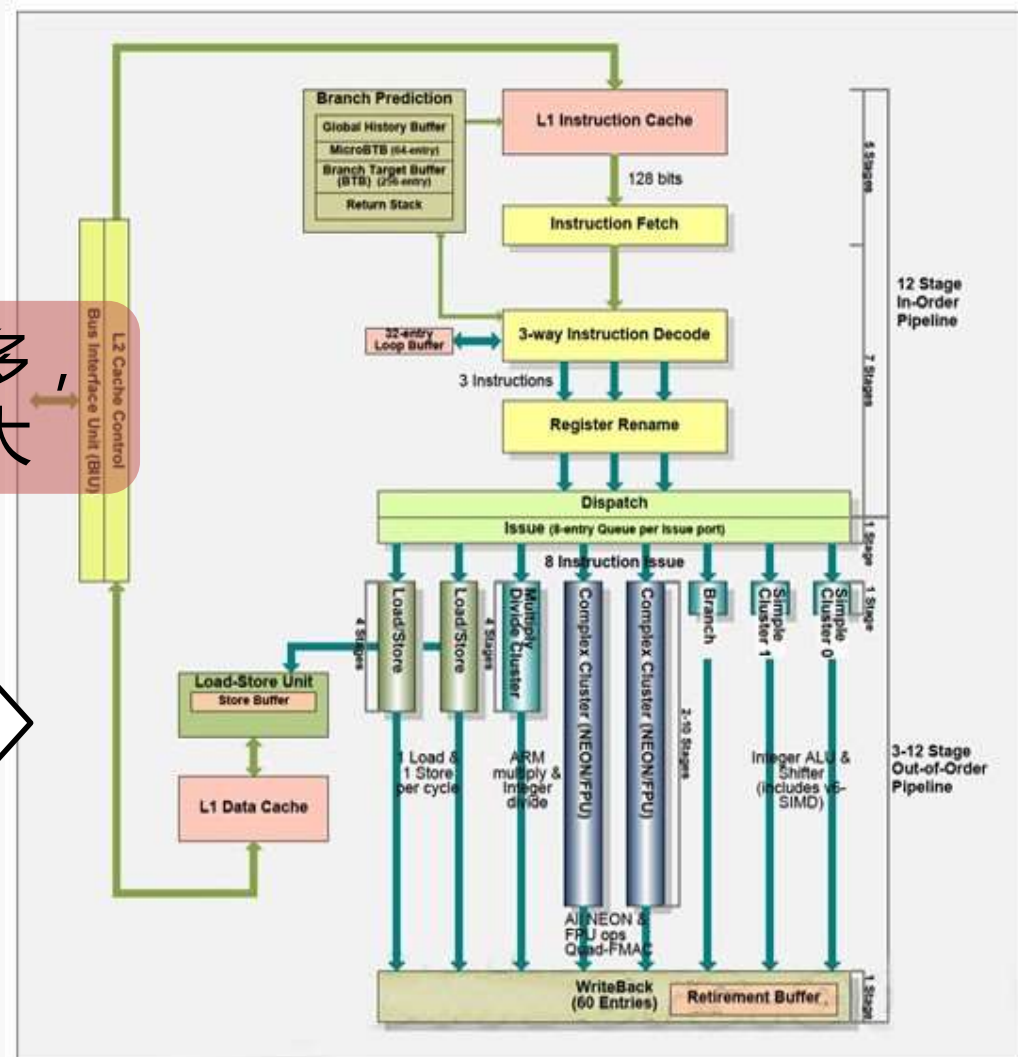
## 🕒 转移指令损失 ( Branch Penalty )

- Alpha 21264 : 转移损失平均为7个周期
- Pentium III : 转移损失平均为10~15个周期
- AMD Athlon K7 : 转移损失10个周期以上



# ARM Cortex-A15

## 3发射 15级流水



# 转移开销



🔍 当执行了转移指令，并确实发生转移时，产生如下的开销，称为“转移开销”

- ① 将按顺序预取的指令废除（即“排空流水线”）
- ② 从转移目标地址重新取指令

🔍 转移开销的构成

- ① “要不要转移？”：转移条件判定引起的开销
- ② “转移到哪里？”：生成目标地址引起的开销

# 转移指令的分类

	无条件转移	条件转移
直接转移	<p><b>x86示例：</b> JMP Target CALL Target</p> <p><b>MIPS示例：</b> <u>j Target</u> jal Target</p>	<p><b>x86示例：</b> JZ Target LOOP Target</p> <p><b>MIPS示例：</b> <u>beq \$t0, \$t1, Target</u> bgez \$t0, Target</p>
间接转移	<p><b>x86示例：</b> JMP DWORD PTR [30H] JMP [EAX] CALL EAX</p> <p><b>MIPS示例：</b> <u>jr \$t0</u></p>	

# 无条件转移

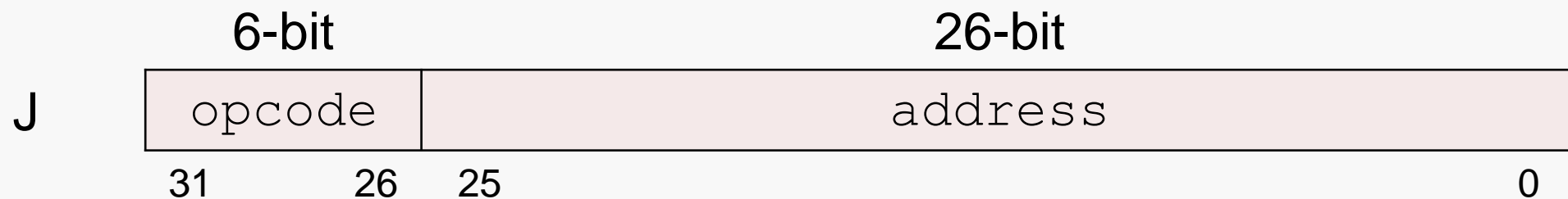


## 直接转移

- $j$  *Target*

## 目标地址计算方法

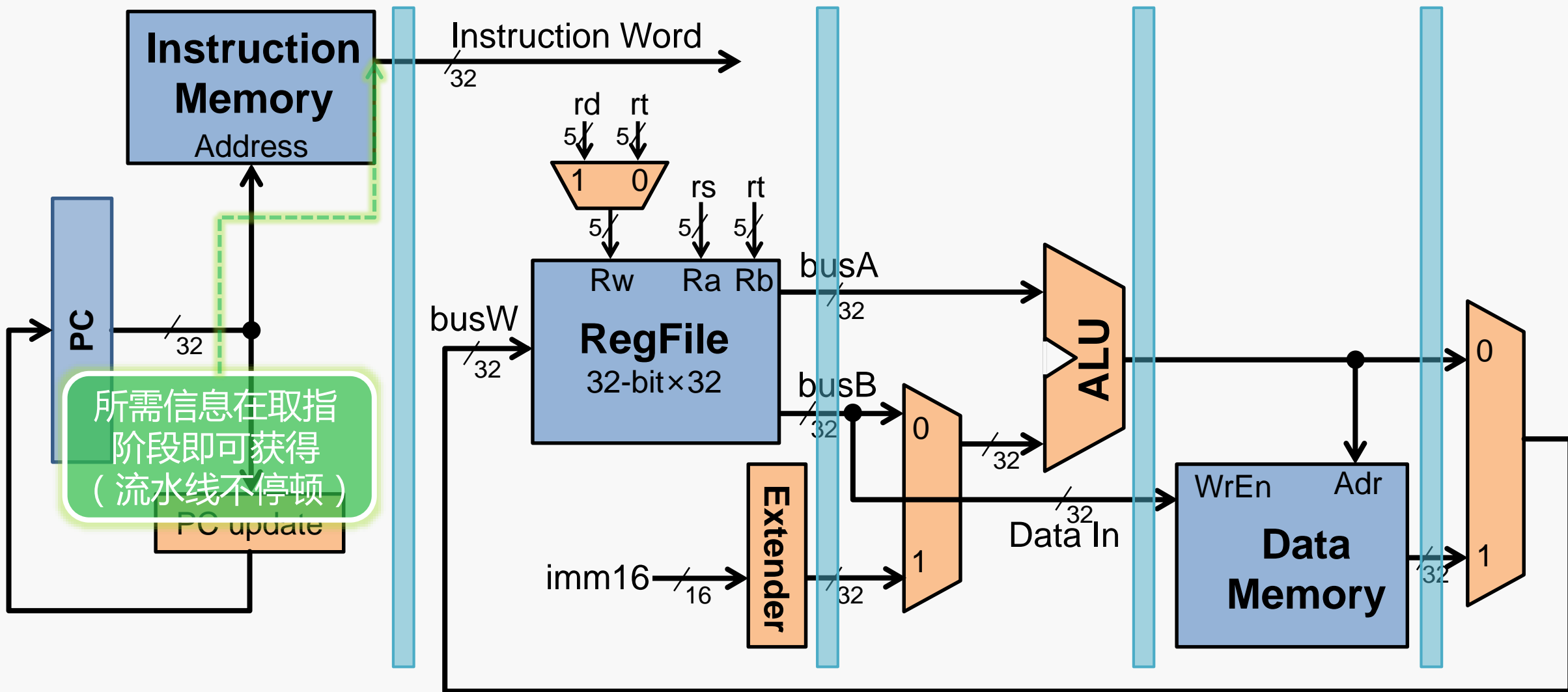
- $\text{New PC} = \{ (\text{PC}+4) [31..28], \text{address}, 00 \}$





# 无条件直接转移 (j Target)

←----- 取指 ----->   ←----- 译码 ----->   ←----- 执行 ----->   ←----- 访存 ----->   ← 写回 -->





# 无条件转移

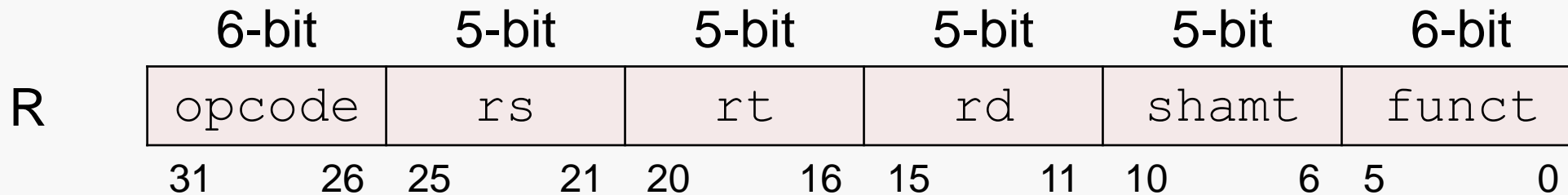


## 间接转移

- `jr rs`

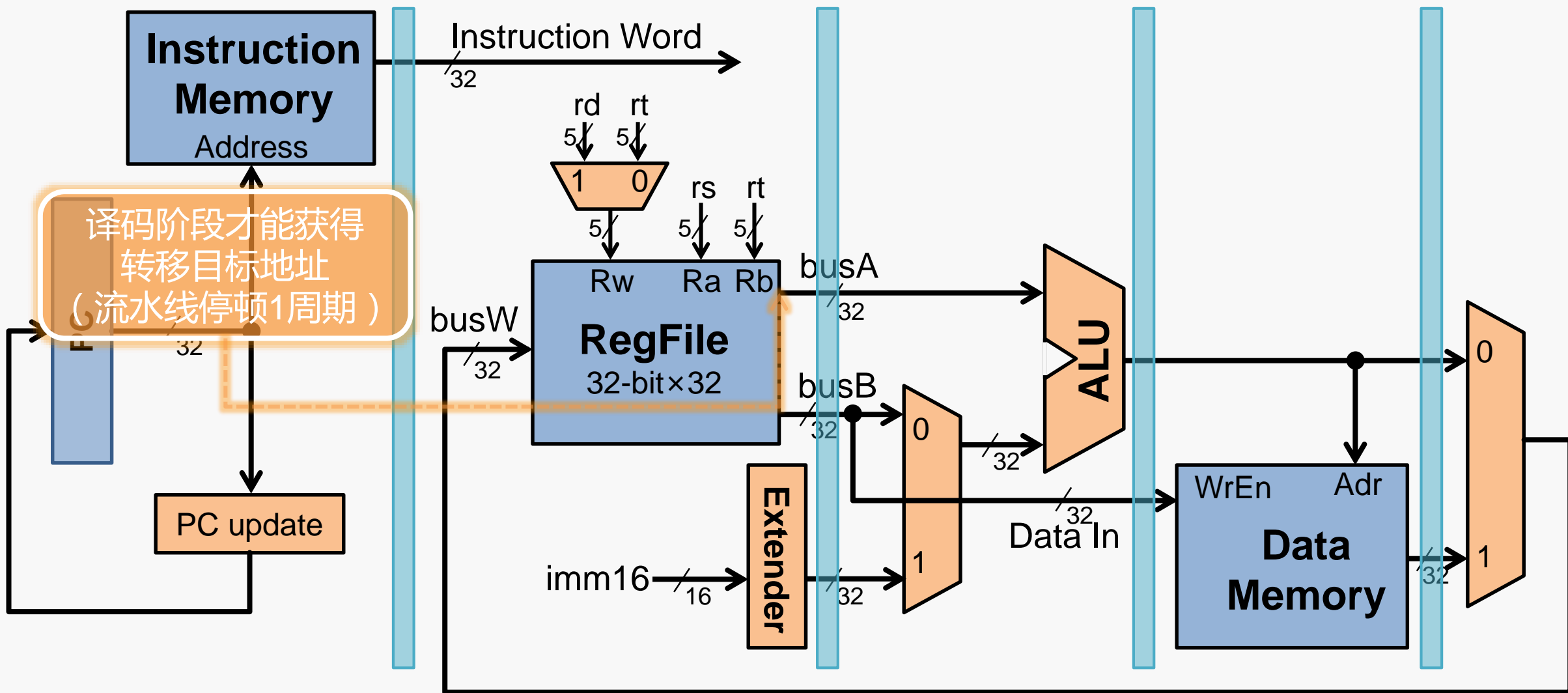
## 目标地址计算方法

- $\text{New PC} = R[\text{rs}]$



# 无条件 间接 转移 ( jr rs )

←----- 取指 ----->   ←----- 译码 ----->   ←----- 执行 ----->   ←----- 访存 ----->   ←----- 写回 ----->



# 条件转移

## 直接转移

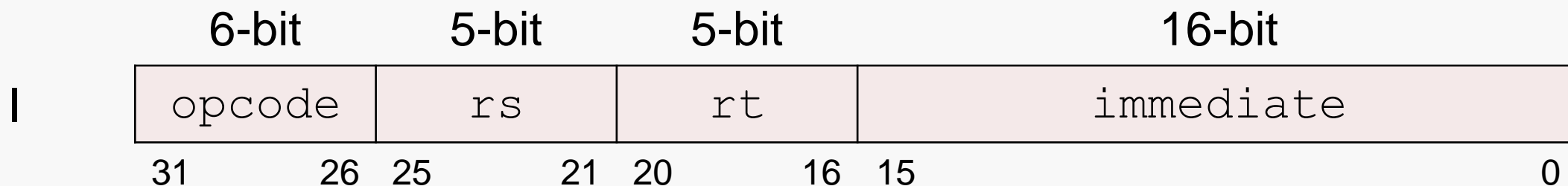
◦ `beq rs, rt, imm16`

## 目标地址计算方法

`if (R[rs] - R[rt] == 0)`

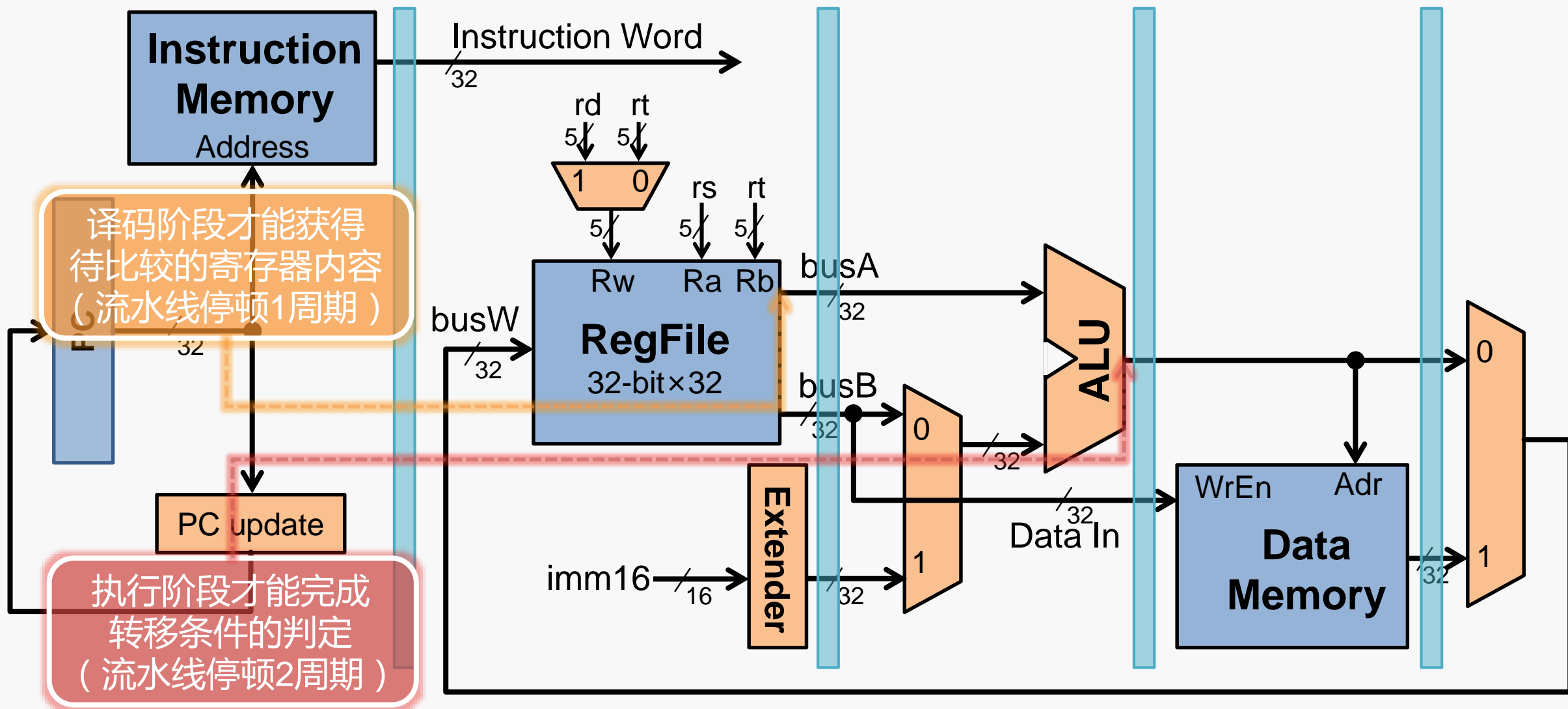
`then New PC = PC + 4 + SignExt[imm16] * 4 ;`

`else New PC = PC + 4 ;`



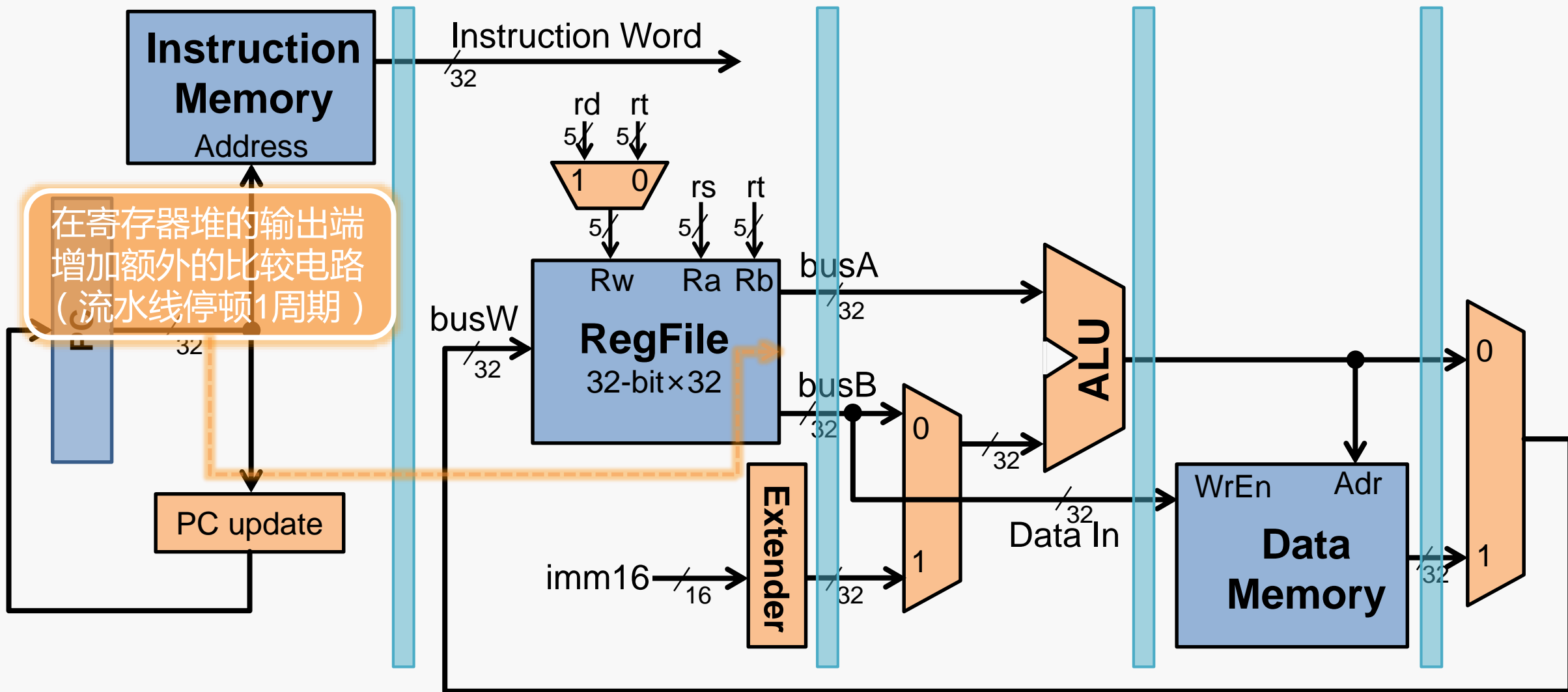
# 条件转移 ( beq rs,rt,imm16 )

←----- 取指 ----->   ←----- 译码 ----->   ←----- 执行 ----->   ←----- 访存 ----->   ←----- 写回 ----->



# 条件转移 (beq rs, rt, imm16)

←----- 取指 ----->   ←----- 译码 ----->   ←----- 执行 ----->   ←----- 访存 ----->   ←----- 写回 ----->




# 控制冒险的影响




- ④ 无条件直接转移 : `j Target`
  - 流水线无停顿
- ④ 无条件间接转移 : `jr rs`
  - 流水线停顿1个周期
- ④ 条件转移 : `beq rs, rt, imm16`
  - 流水线停顿1个周期

# 延迟转移技术

流水线不停顿！



```
xor    $s1, $s2, $s3
addi   $t1, $t3, 1
subi   $t2, $t4, 2
beq    $t1, $t2, Next
slt    $s4, $s5, -50
...
Next:  ...
```



```
addi   $t1, $t3, 1
subi   $t2, $t4, 2
beq    $t1, $t2, Next
xor    $s1, $s2, $s3
slt    $s4, $s5, -50
...
Next:  ...
```



## 本节小结



# 控制冒险的处理

北京大学·慕课  
计算机组成  
制作人：陆俊林

