单周期MIPS32处理器设计

Single-Cycle MIPS32 Processor Design

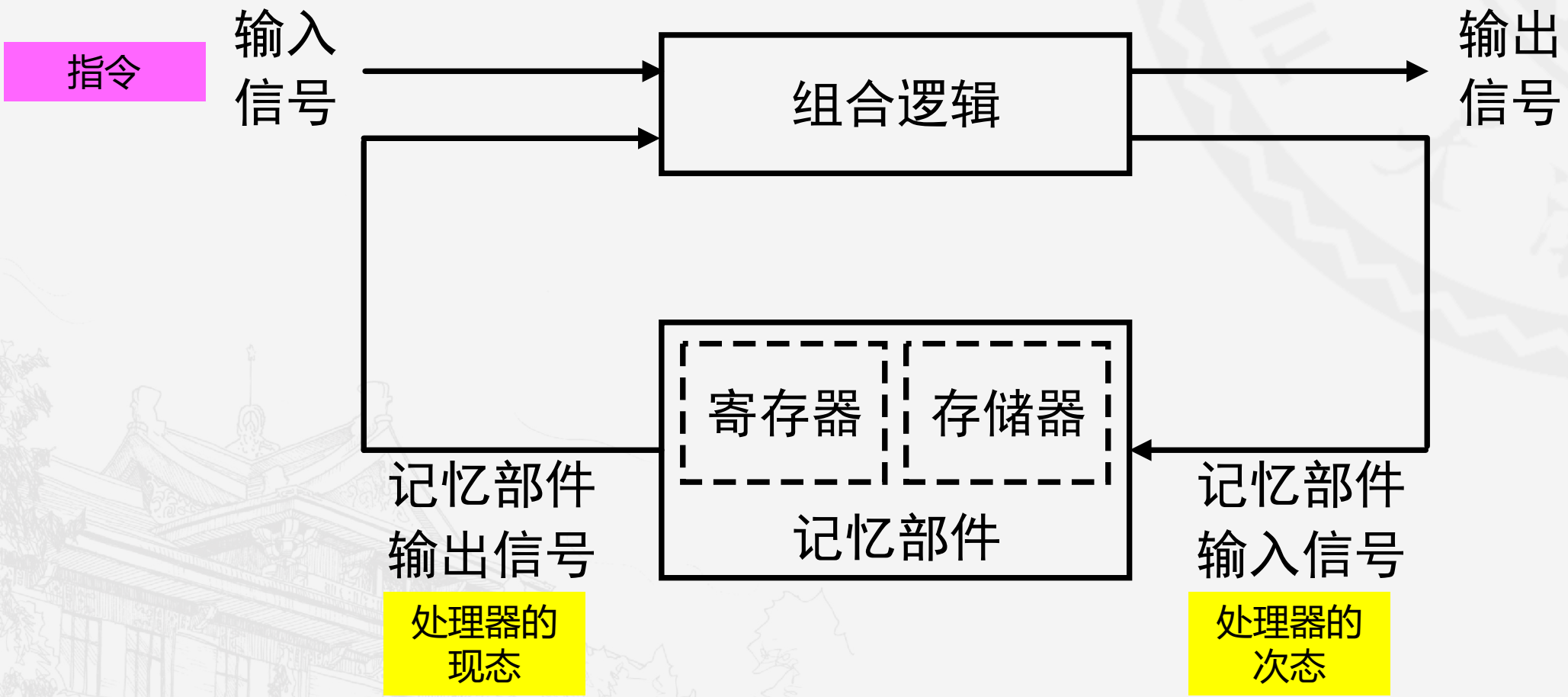
设计要求

- 单周期MIPS32处理器
  - 每条指令均在一个时钟周期完成
  - 32个32位寄存器，哈佛结构，小端模式，支持23条指令
- 数据通路 + 控制通路
  - 数据通路：完成对指令中操作数的运算、存储等处理工作
  - 控制通路：从数据通路中接收指令，并对其进行翻译以告知数据通路如何处理
- 处理器设计相当于在各个记忆部件之间添加组合逻辑电路，在控制单元的控制下根据当前电路的状态计算出电路的新状态

单周期MIPS32处理器的设计

Single-Cycle MIPS Processor Design

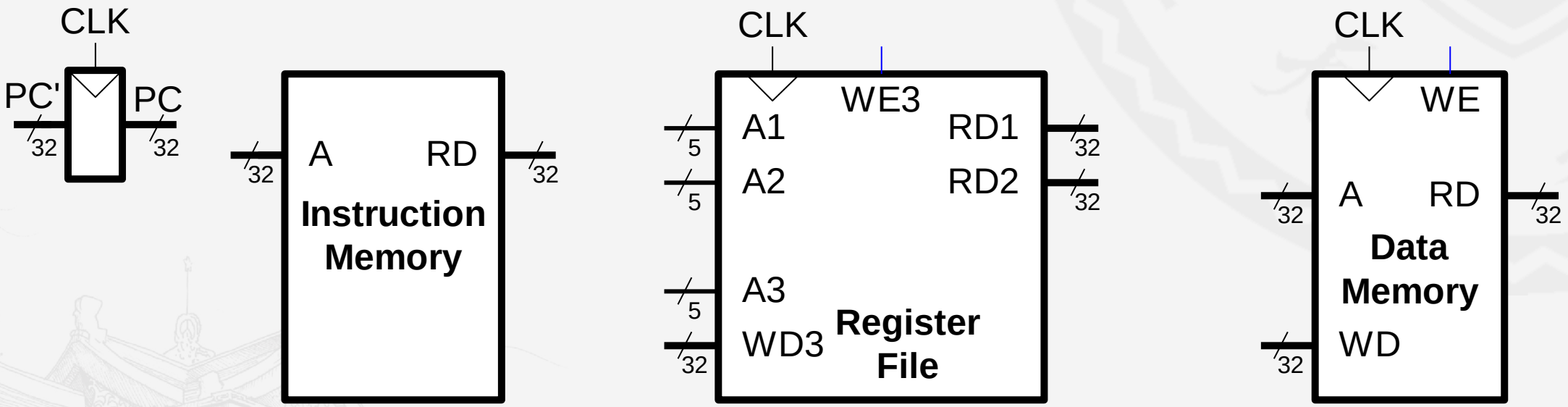
MiniMIPS32处理器概念模型



单周期MIPS32处理器的设计

Single-Cycle MIPS Processor Design

记忆部件

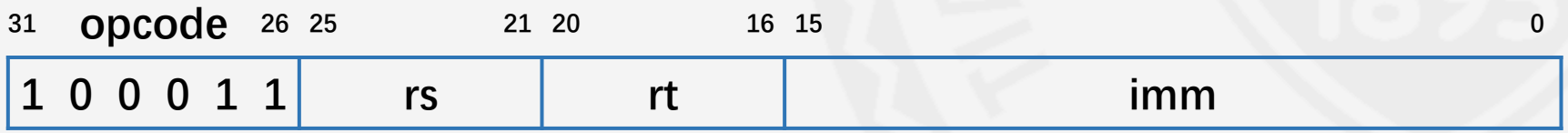


单周期MIPS32处理器的设计

Single-Cycle MIPS Processor Design

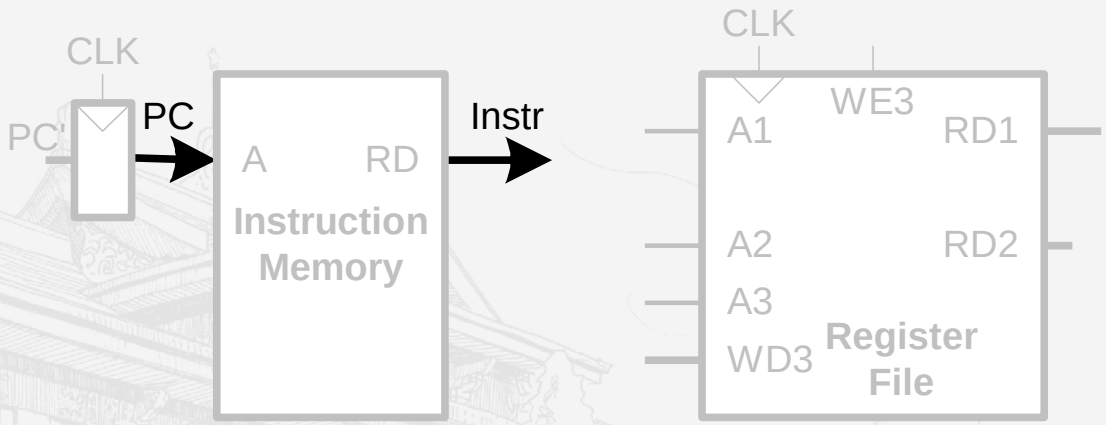
lw指令数据通路（I类型指令）

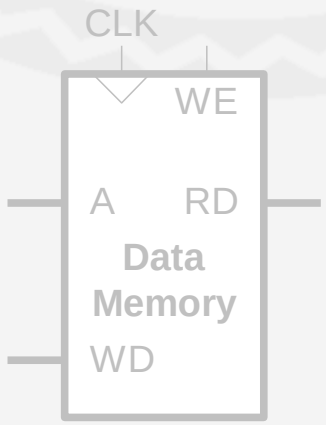
```
lw rt, imm(rs)
```



imm需要符号扩展为32位有符号数

Step 1 取指令

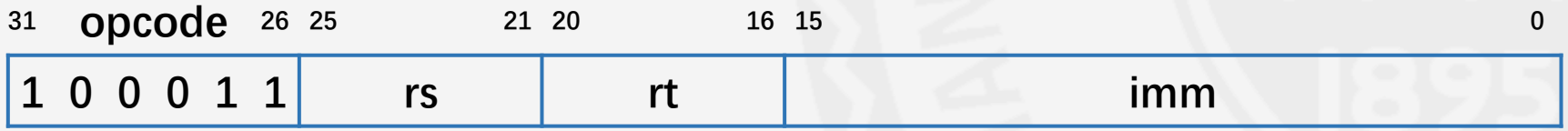




单周期MIPS32处理器的设计

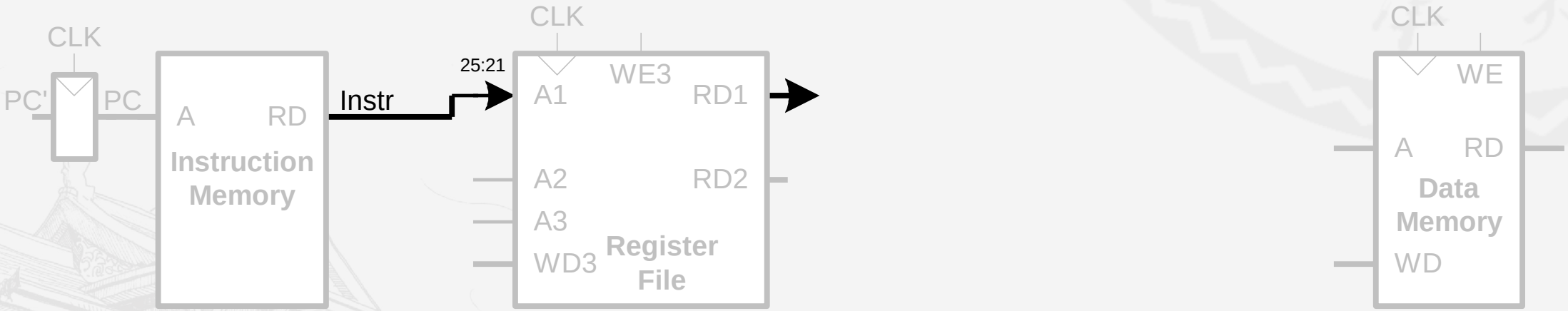
Single-Cycle MIPS Processor Design

`lw rt, imm(rs)`



imm需要符号扩展为32位有符号数

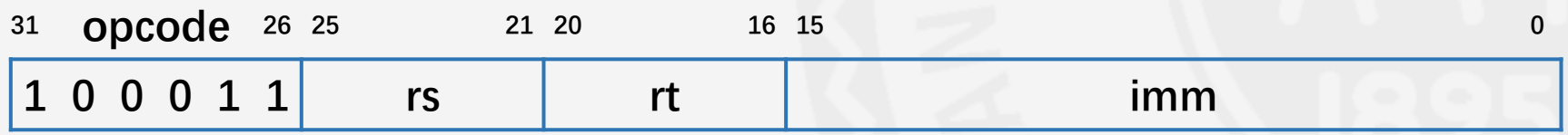
Step 2-1 译码阶段 —— 从寄存器取操作数



单周期MIPS32处理器的设计

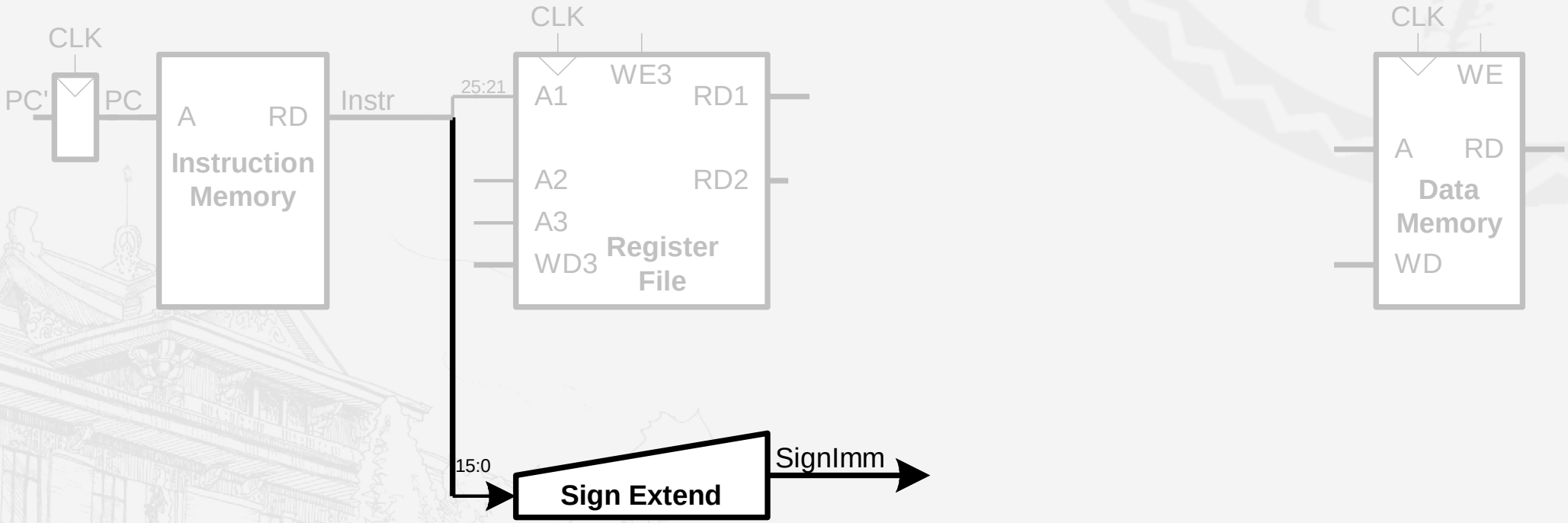
Single-Cycle MIPS Processor Design

```
lw rt, imm(rs)
```



imm需要符号扩展为32位有符号数

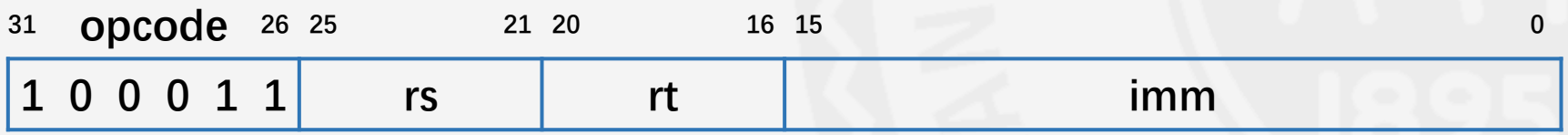
Step 2-2 译码阶段 — —另一个操作数进行符号扩展



单周期MIPS32处理器的设计

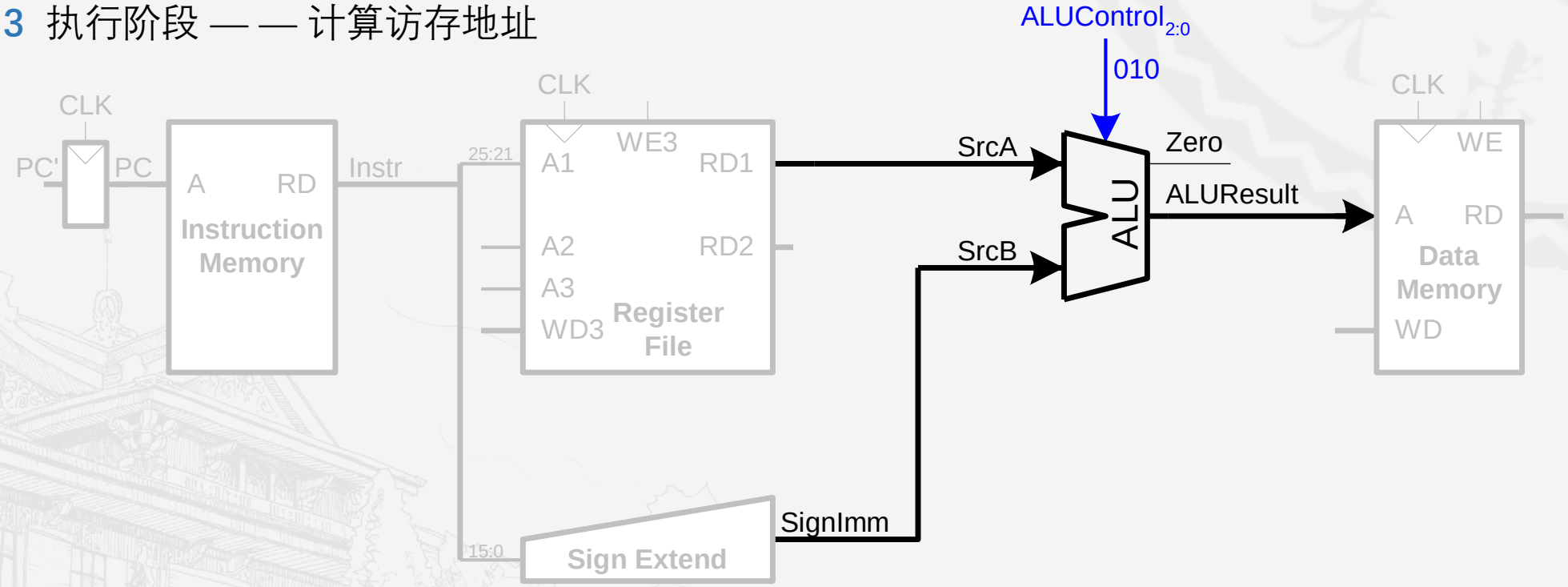
Single-Cycle MIPS Processor Design

`lw rt, imm(rs)`



imm需要符号扩展为32位有符号数

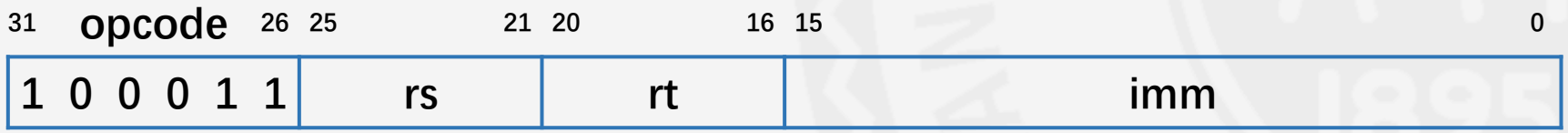
Step 3 执行阶段 —— 计算访存地址



单周期MIPS32处理器的设计

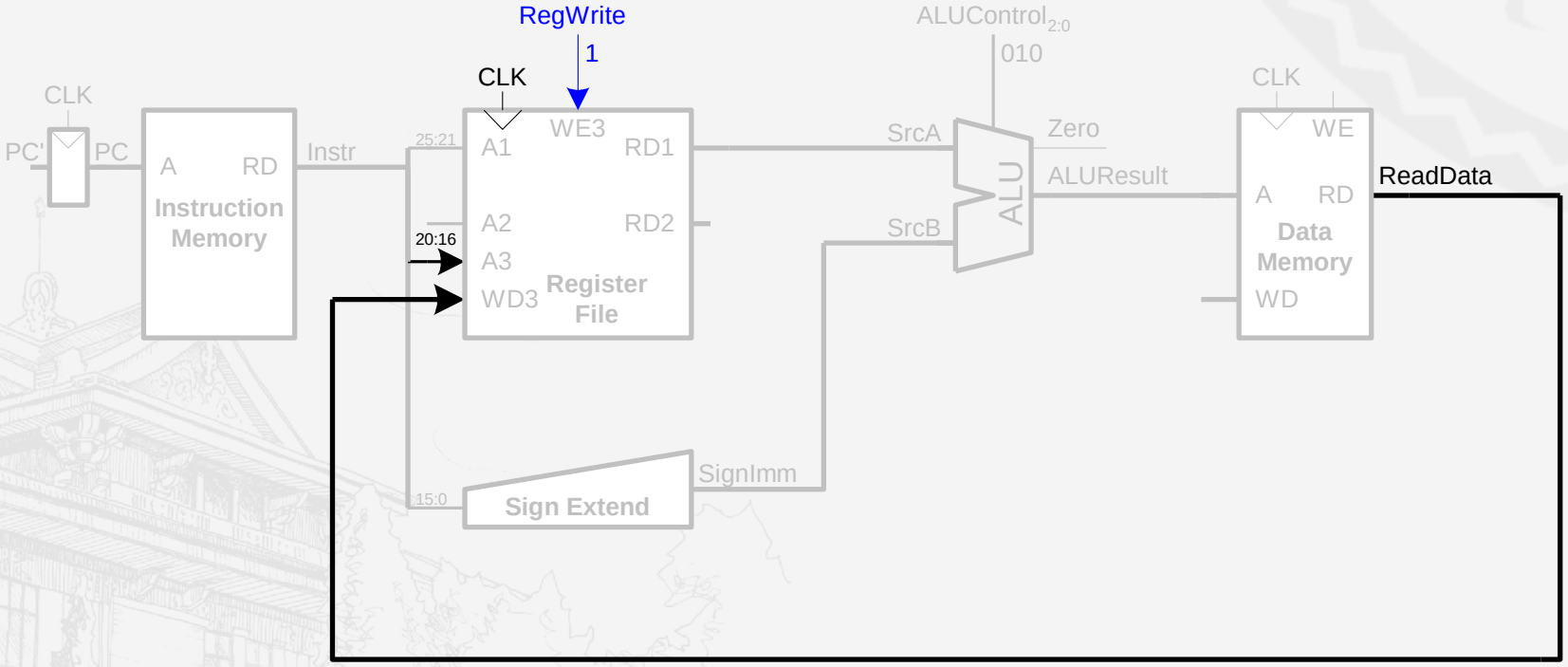
Single-Cycle MIPS Processor Design

```
lw rt, imm(rs)
```



imm需要符号扩展为32位有符号数

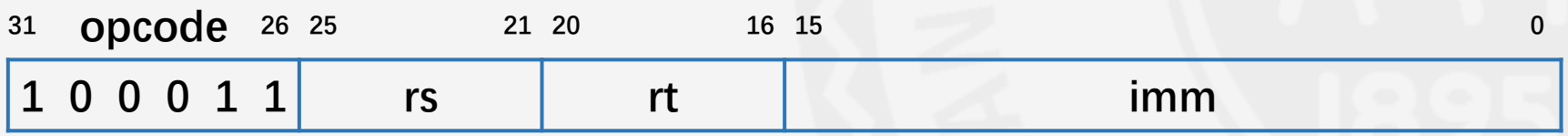
Step 4 访存阶段/写回阶段 — —从数据存储器取回数据，写入寄存器文件



单周期MIPS32处理器的设计

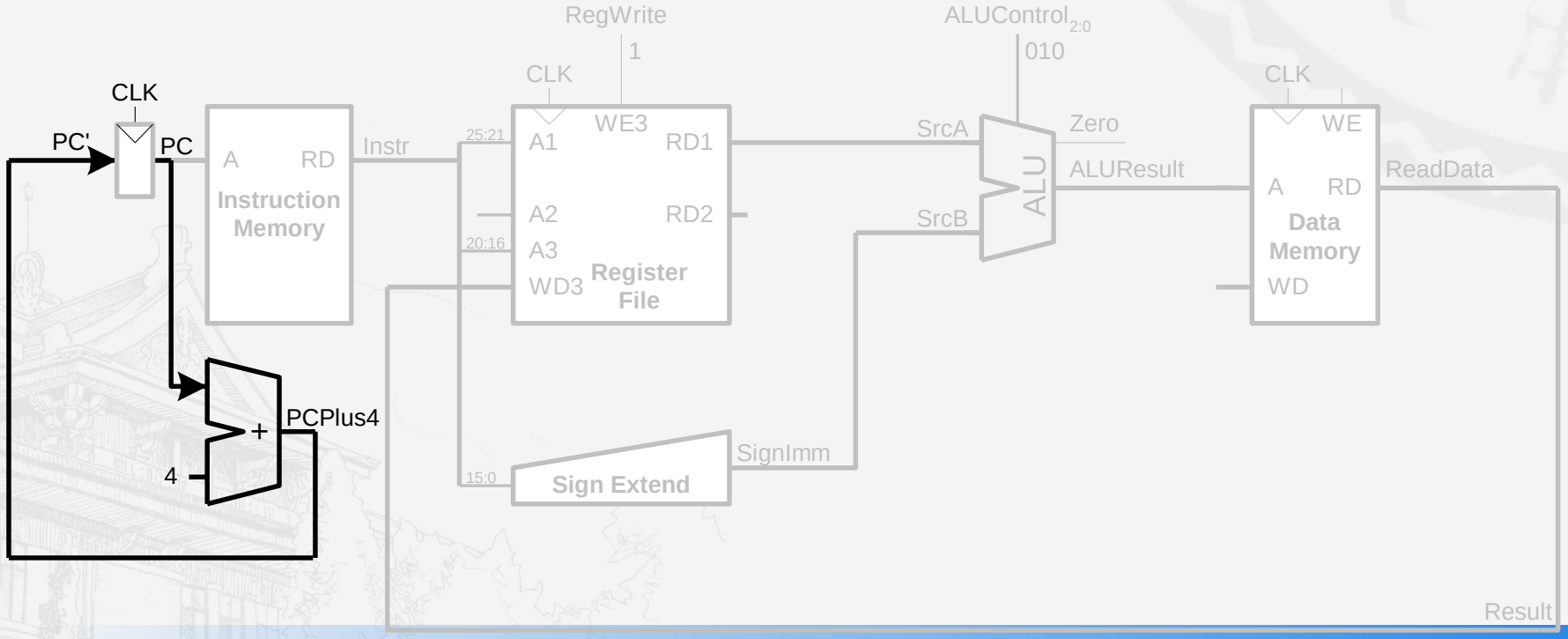
Single-Cycle MIPS Processor Design

`lw rt, imm(rs)`



imm需要符号扩展为32位有符号数

Step 5 PC — — 计算下一条指令的地址

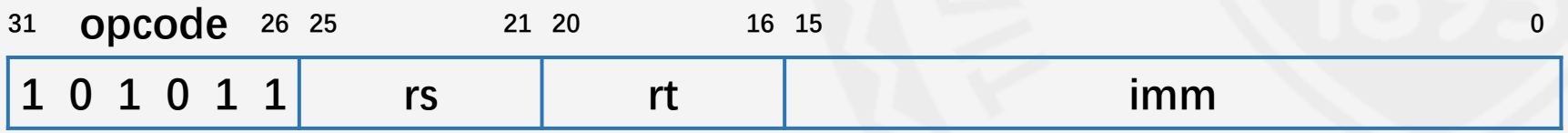


单周期MIPS32处理器的设计

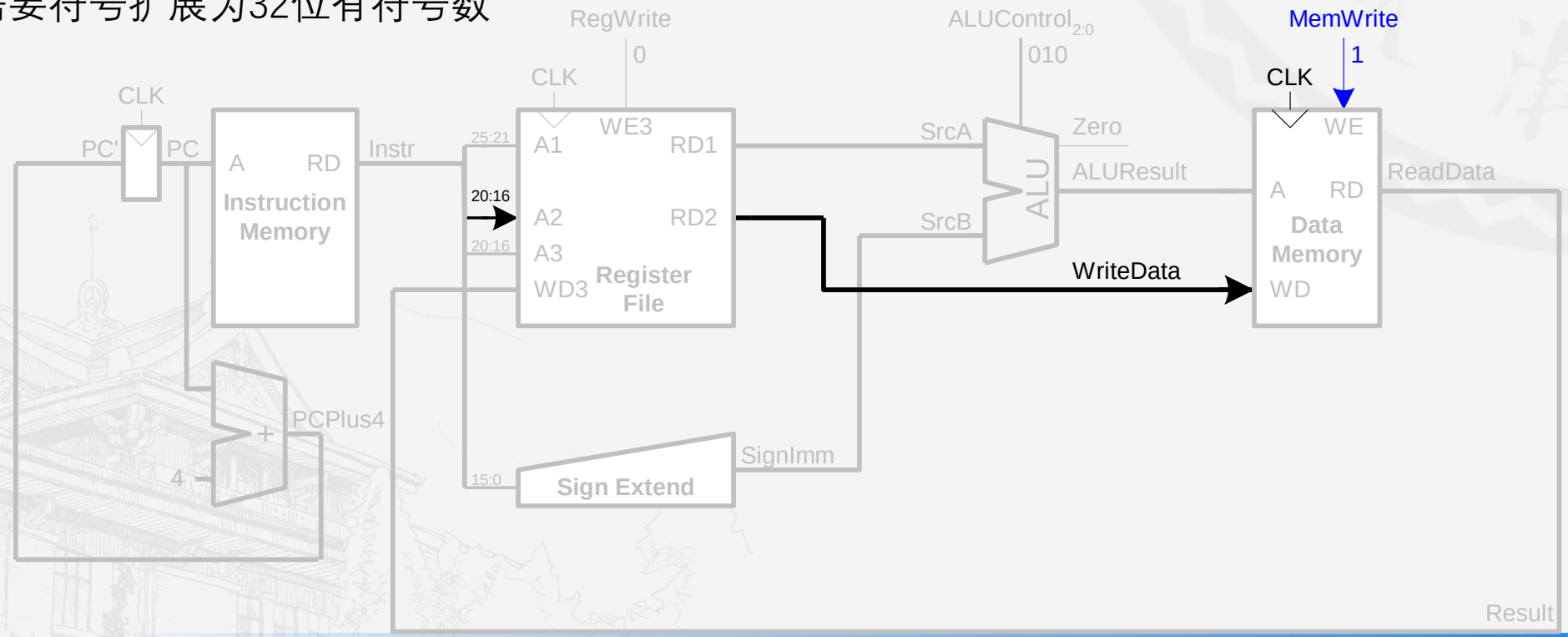
Single-Cycle MIPS Processor Design

sw指令数据通路（I类型指令）

```
sw rt, imm(rs)
```



imm需要符号扩展为32位有符号数

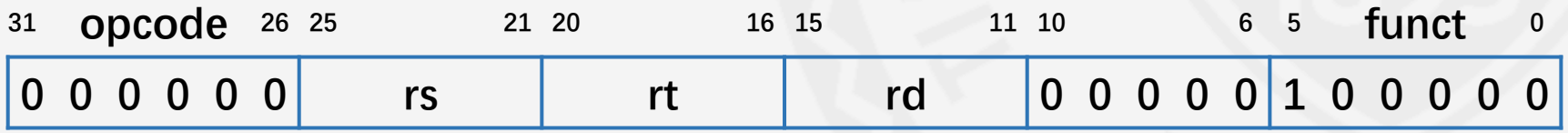


单周期MIPS32处理器的设计

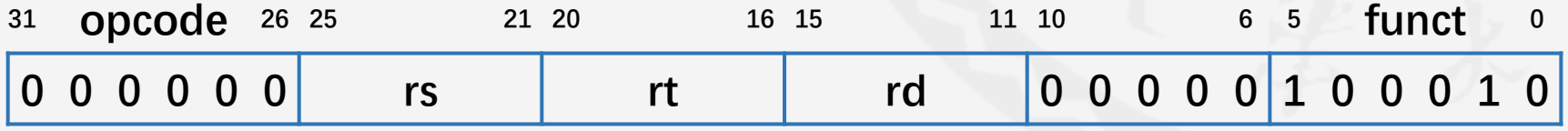
Single-Cycle MIPS Processor Design

R类型指令回顾

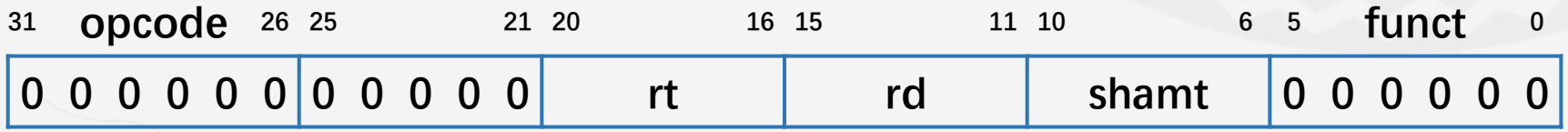
`add rd, rs, rt`



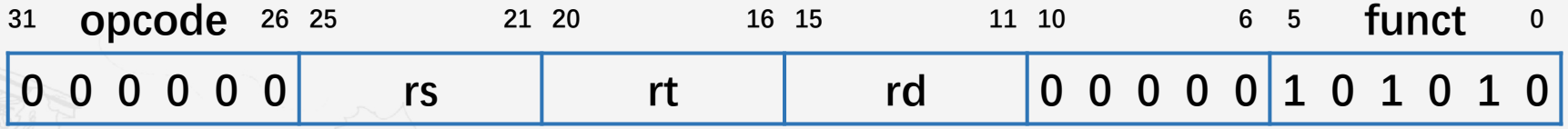
`sub rd, rs, rt`



`sll rd, rt, shamt`



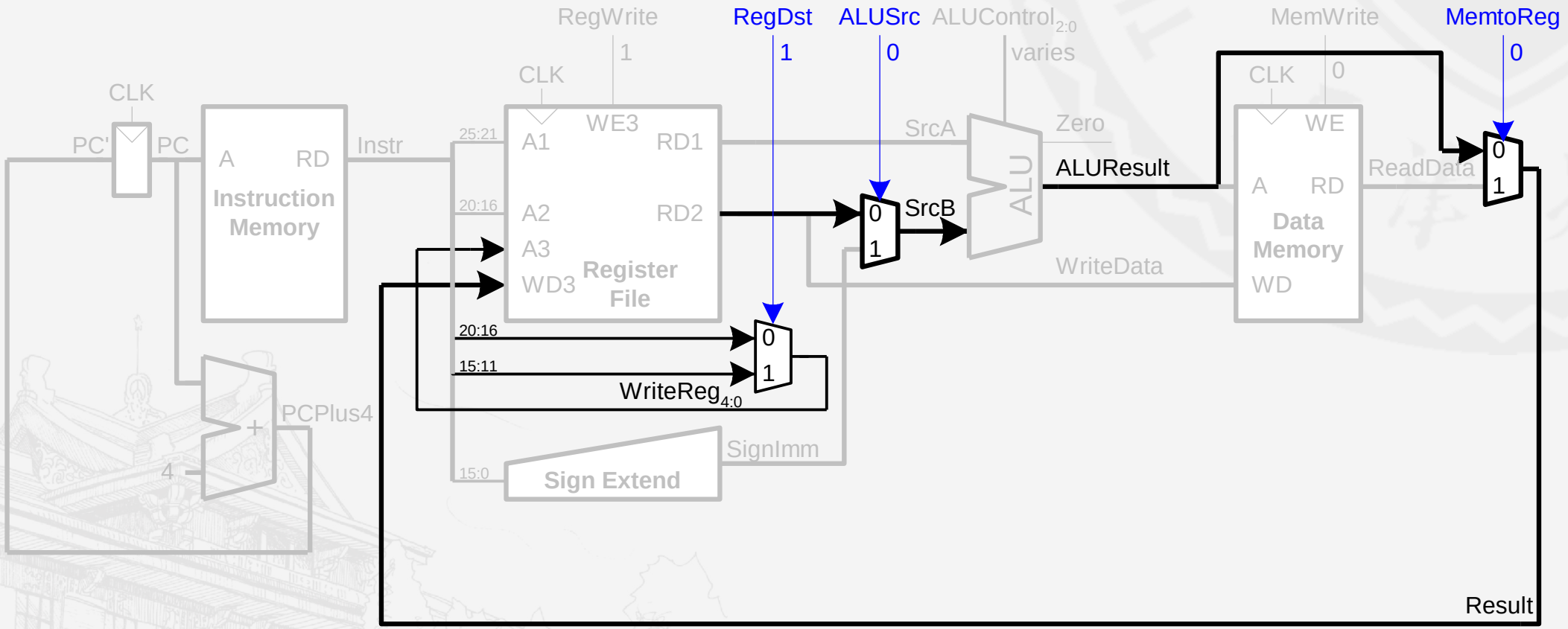
`slt rd, rs, rt`



单周期MIPS32处理器的设计

Single-Cycle MIPS Processor Design

R类型指令数据通路



单周期MIPS32处理器的设计

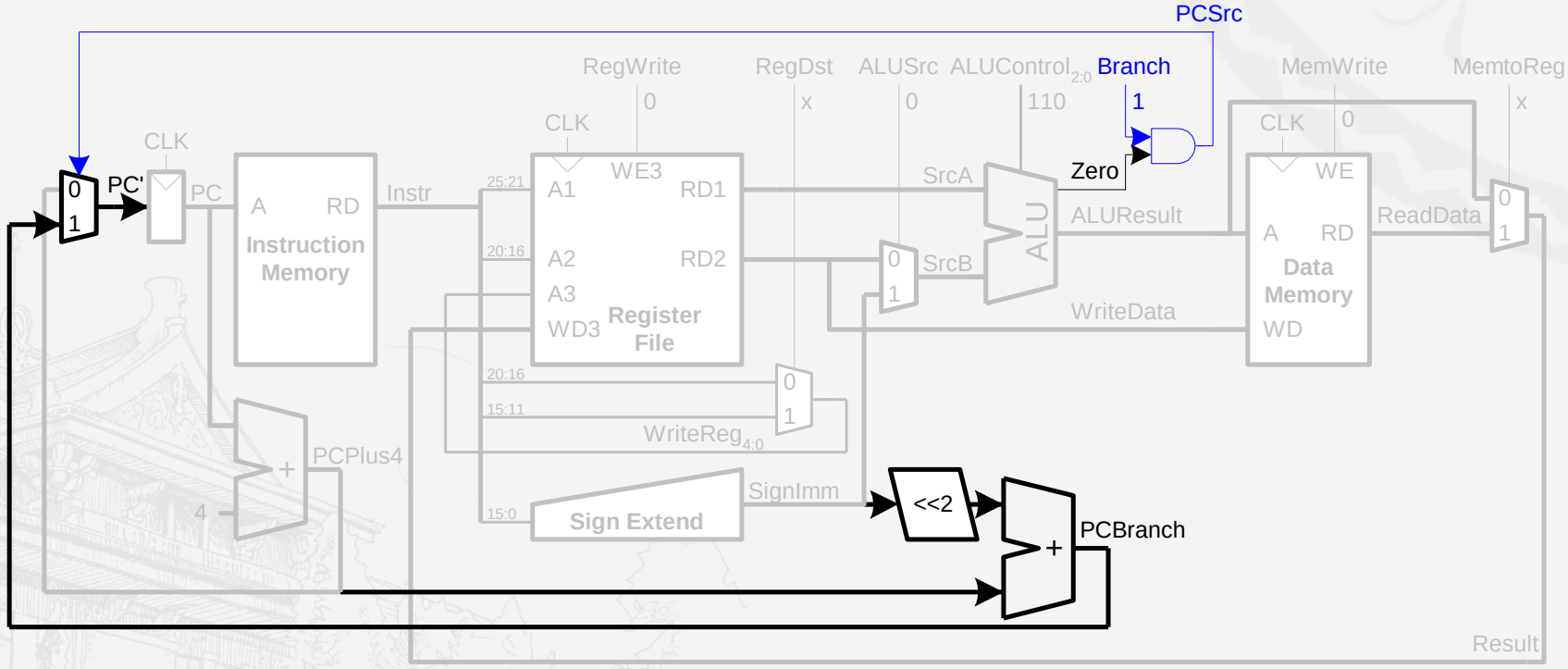
Single-Cycle MIPS Processor Design

beq指令数据通路（I类型指令）

```
beq rs, rt, label
```

31	opcode	26	25	21	20	16	15	0
0 0 0 1 0 0			rs		rt		imm	

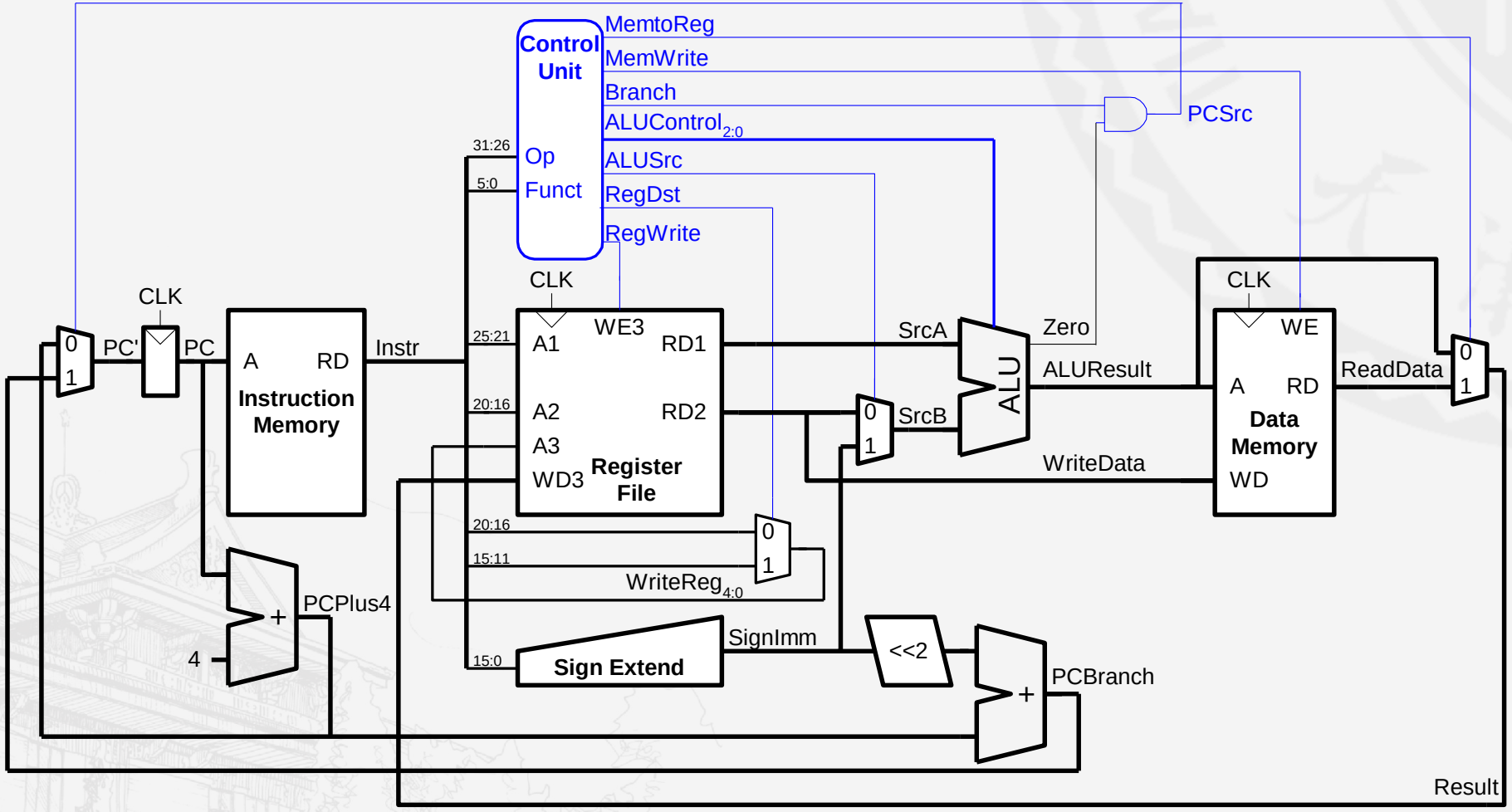
当rs==rt时，进行跳转，跳转的目标地址为 PC + 4 + (SignImm <<2)，SignImm为imm符号扩展为32位的有符号数



单周期MIPS32处理器的设计

Single-Cycle MIPS Processor Design

控制通路

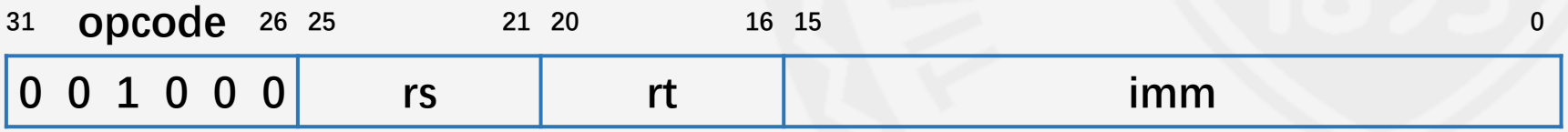


单周期MIPS32处理器的设计

Single-Cycle MIPS Processor Design

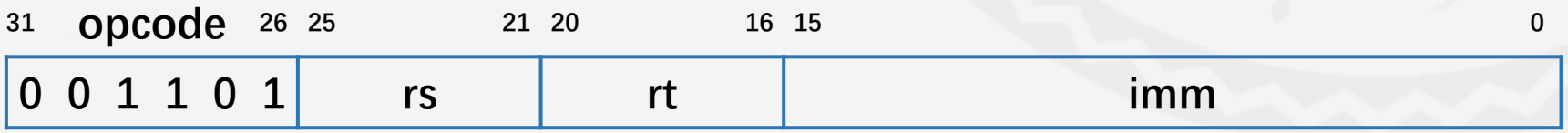
I类型指令

`addi rt, rs, imm`



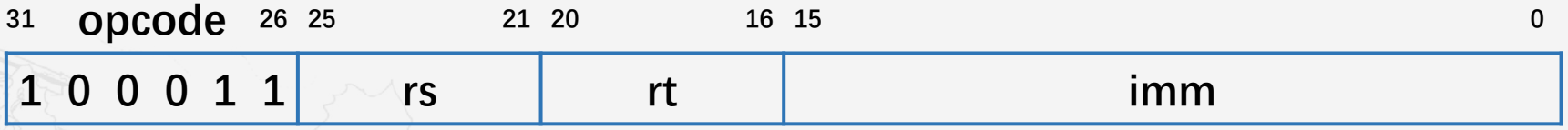
imm需要符号扩展为32位有符号数

`ori rt, rs, imm`



imm需要符号扩展为32位有符号数

`lw rt, imm(rs)`

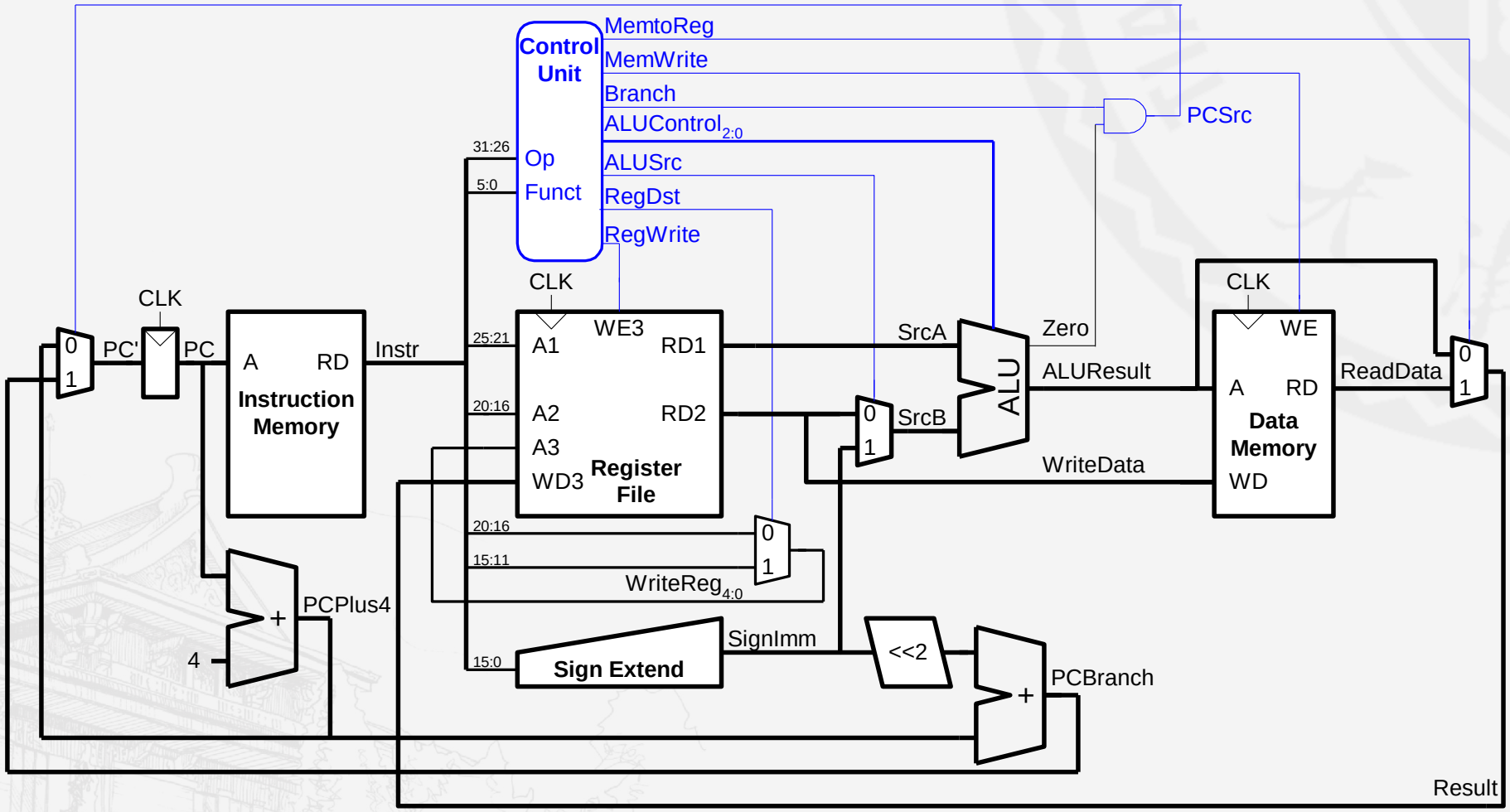


imm需要符号扩展为32位有符号数

单周期MIPS32处理器的设计

Single-Cycle MIPS Processor Design

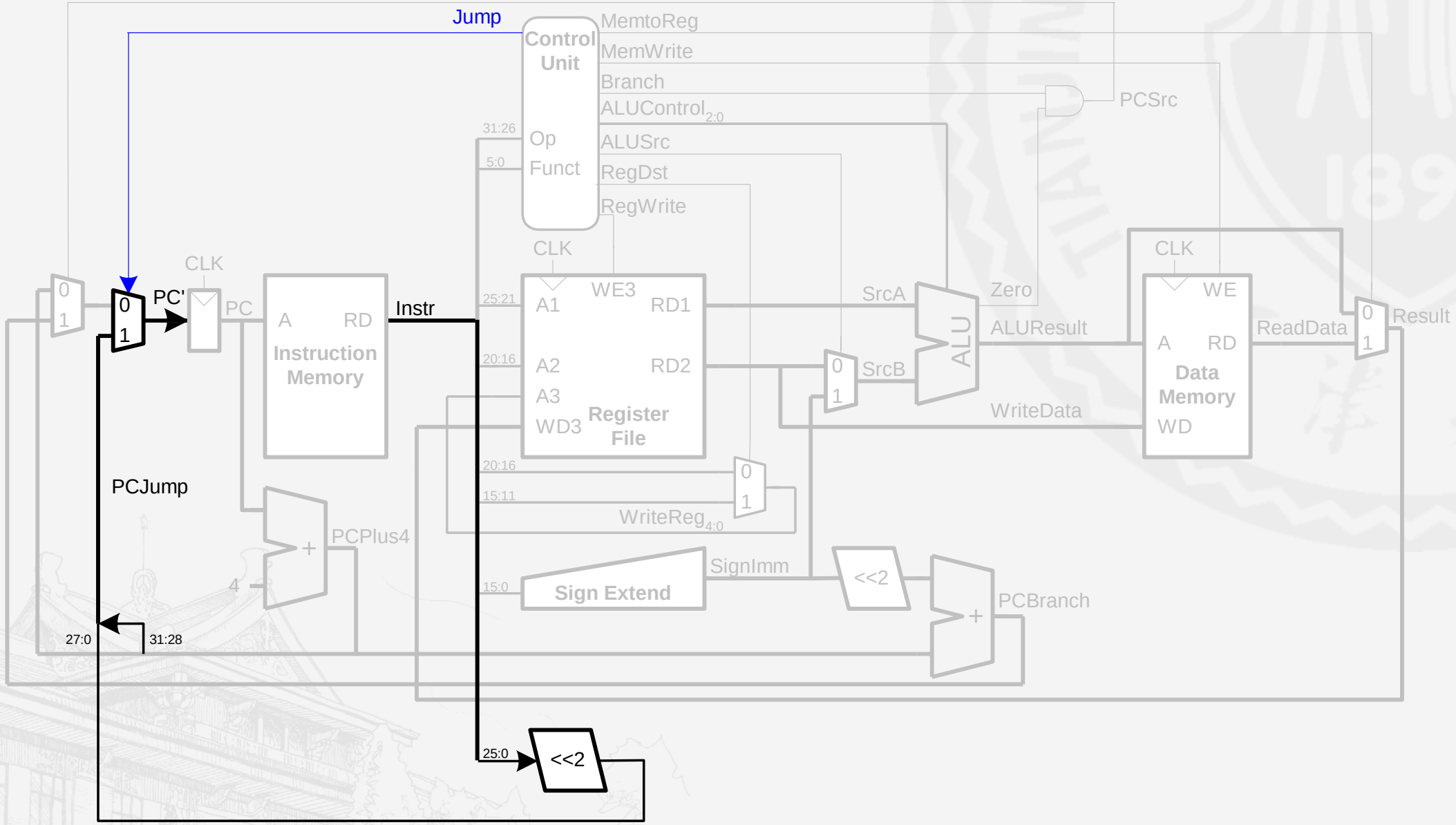
addi/ori指令的实现:数据通路没有任何变化



J类型指令

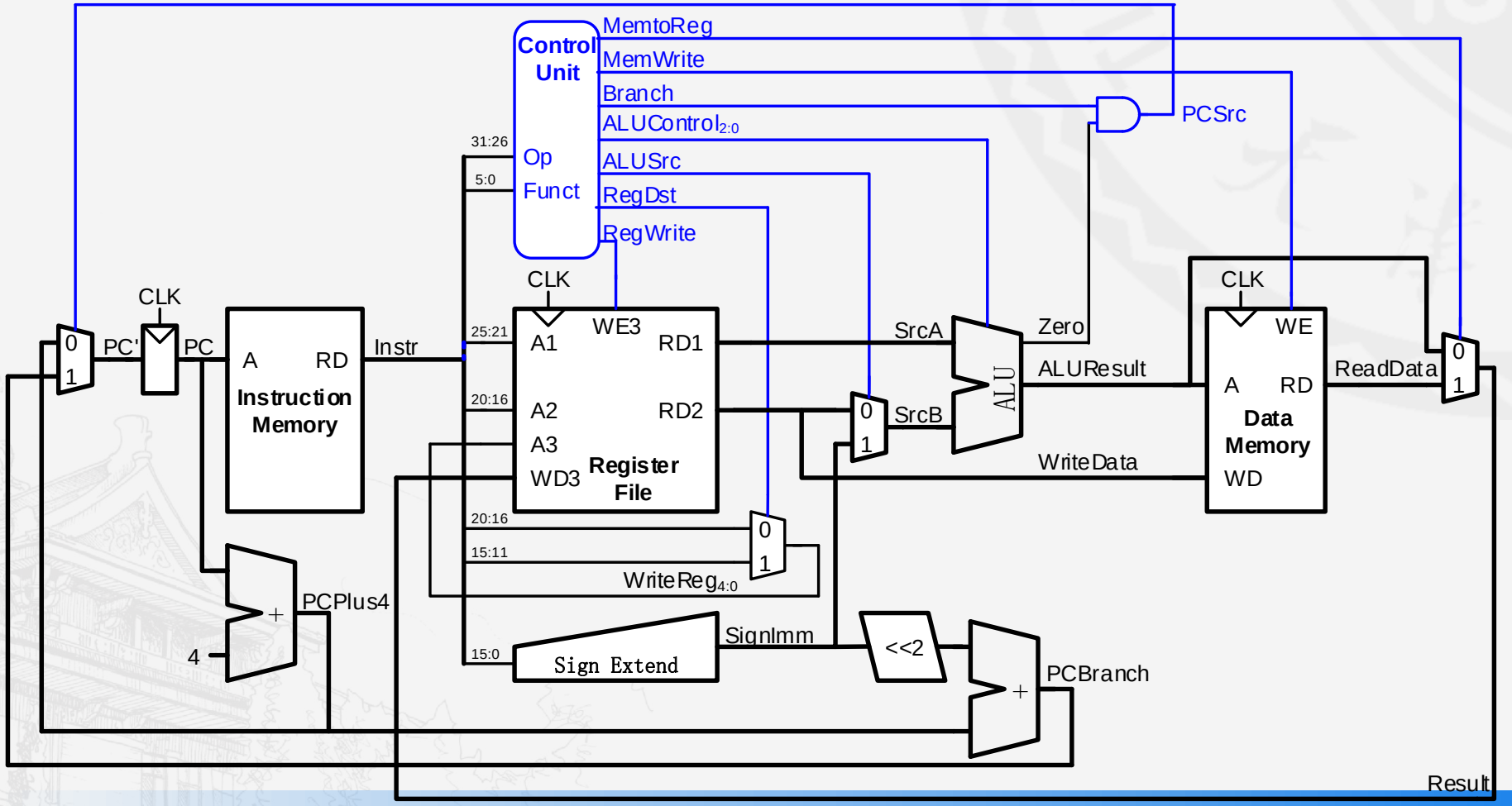
跳转的目标地址为 { (PC+4)[31:28], addr, 2'b0 }

跳转的目标地址为 { (PC+4)[31:28], addr, 2'b0 }



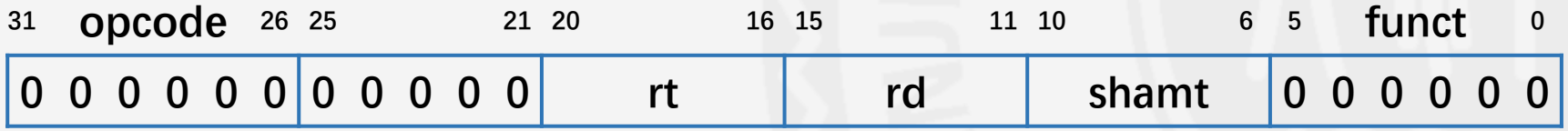
控制器逻辑

Instruction	$Op_{5:0}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$	Jump
R-type	000000	1	1	0	0	0	0	由funct决定	0
lw	100011	1	0	1	0	0	1	加法	0
sw	101011	0	X	1	0	1	X	加法	0
beq	000100	0	X	0	1	0	X	减法	0
j	000010	0	X	X	X	0	X	XX	1

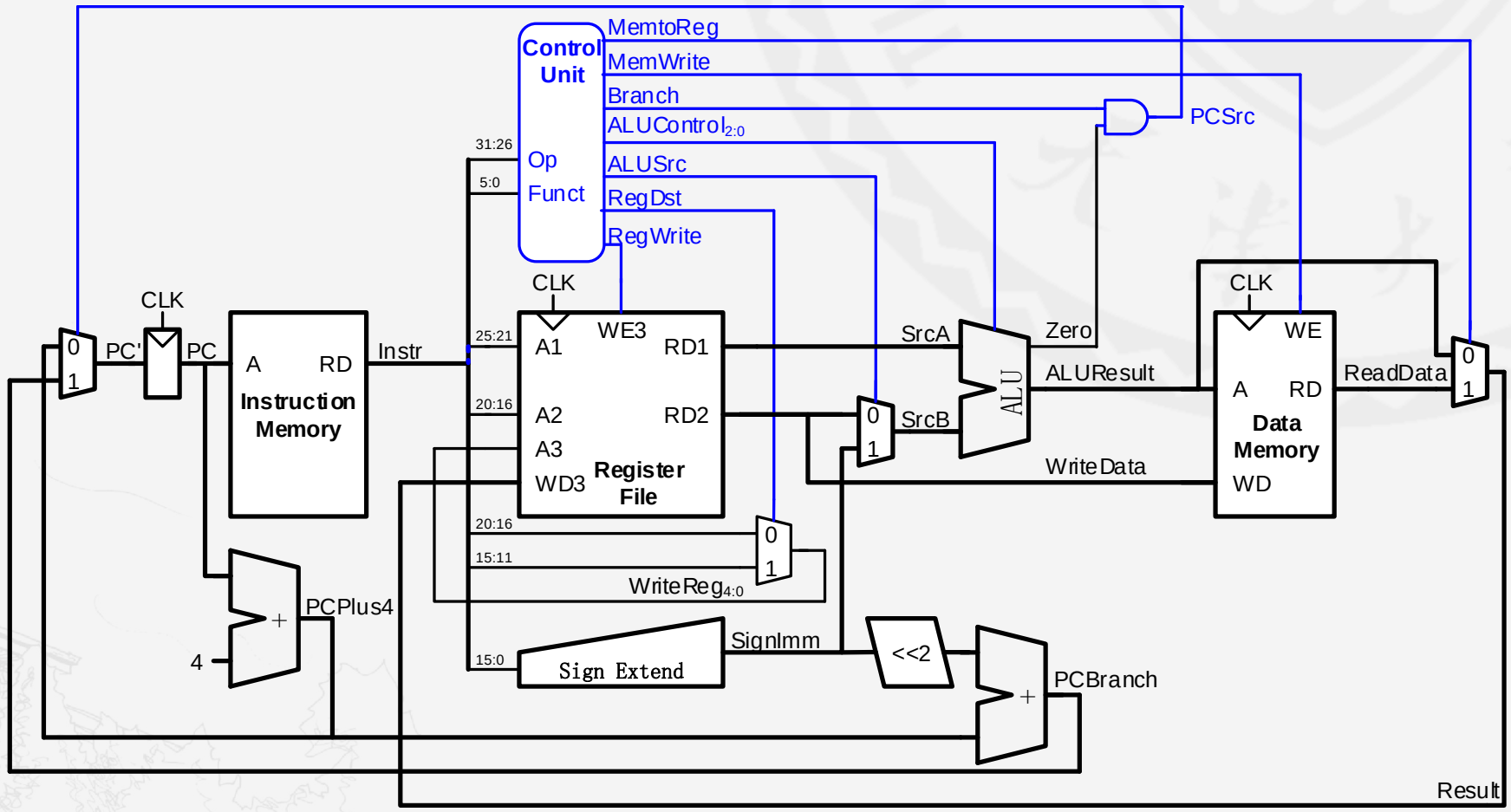


问题：sll指令如何实现?

```
sll rd, rt, shamt
```




```
[rd] = [rt] << shamt
```



单周期MIPS32处理器的设计

Single-Cycle MIPS Processor Design

实验要求

- 支持 10 条指令：
  - lw
  - sw
  - lui
  - ori
  - addiu
  - addu
  - slt
  - beq
  - bne
  - j
  - 每条指令均在一个时钟周期完成
- 采用哈佛结构