

第六讲 MIPS处理器设计

- 一. 处理器设计概述
- 二. MIPS模型机
- 三. MIPS单周期处理器设计
 - 1. 单周期数据通路设计
 - 2. 单周期控制器设计
 - 3. 单周期性能分析
- 四. MIPS多周期处理器设计**
 - 1. 多周期数据通路设计**
 - 2. 多周期控制器设计**
 - 3. 多周期性能分析**

4.1 MIPS 多周期数据通路设计

❖ 为什么不使用单周期实现方式？

- 单周期设计中，时钟周期对所有指令等长
- 而时钟周期由计算机中可能的最长路径决定，如：取数指令
- 但某些指令类型本来可以在更短时间内完成，如：跳转指令

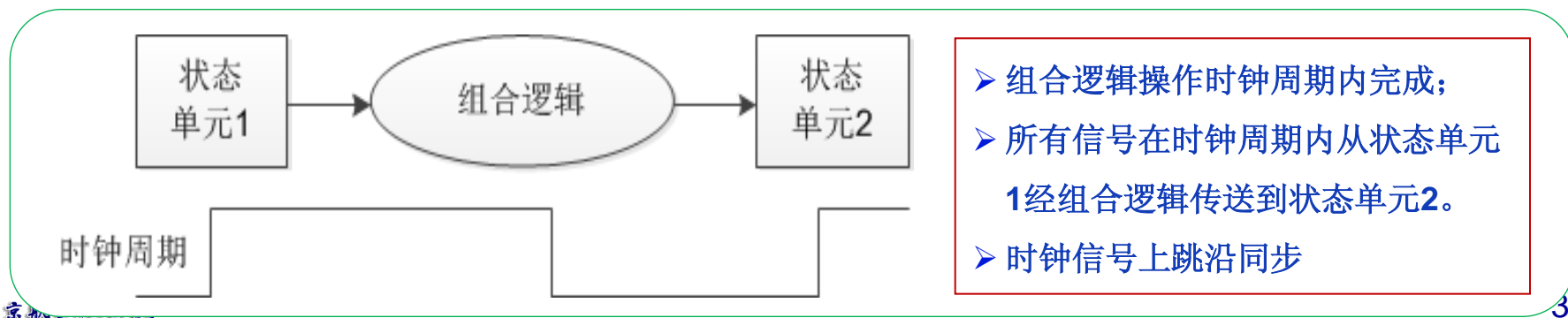
❖ 多周期方案

- 将指令执行分解为多个步骤，每一步骤一个时钟周期，则指令执行周期为多个时钟周期，不同指令的指令周期包含时钟周期数不一样
- 优点：
 - **提高性能**：不同指令的执行占用不同的时钟周期数
 - **降低成本**：一个功能单元可以在一条指令执行过程中使用多次，只要是在不同时钟周期中（这种共享可减少所需的硬件数量）

4.1 MIPS多周期数据通路设计

❖ 多周期数据通路设计总体考虑

- **普林斯顿结构**：指令和数据使用同一个存储器
- **共享一个ALU**：
 - R型指令算术逻辑运算、取指令后形成PC+4新值、及Beq指令转向地址计算（ $PC + \text{Signext}(\text{imm16}) \ll 2$ ），都在ALU中完成
- **时钟同步方法**：
 - 一个时钟周期内，信号总是从一个状态单元，经过组合逻辑处理后，传送到另一个状态单元
 - 指令每一步的执行，总是从前一个状态单元接收输入，经过功能单元处理，在下一个时钟周期触发沿将结果写入下一个状态单元
 - 因此，数据通路中需要增加一个或多个寄存器，以保存指令各执行步骤形成的结果（输出值），以便在指令的后续时钟周期内继续使用



4.1 MIPS多周期数据通路设计

1. R型指令多周期分析

◆ $R[rd] \leftarrow R[rs] \text{ op } R[rt]$

Op (31-26)	Rs (25-21)	Rt (20-16)	Rd (15-11)	Shamt (10-6)	Func (5-0)
---------------	---------------	---------------	---------------	-----------------	---------------

步骤	RTL描述	执行部件	备注
取指令	$IR \leftarrow M[PC]$ $PC \leftarrow PC + 4$	存储器 ALU	增加一个寄存器保存指令: IR
读寄存器	$A \leftarrow R[IR[25:21]]$ $B \leftarrow R[IR[20:16]]$	寄存器堆	增加两个寄存器保存读取的数据R[rs]和R[rt]: A、B
计算	$ALUOut \leftarrow A \text{ op } B$	ALU	增加一个寄存器保存ALU结果: ALUOut
写寄存器	$R[IR[15:11]] \leftarrow ALUOut$	寄存器堆	完成寄存器数据写入

4.1 MIPS多周期数据通路设计

2. Lw指令多周期分析

Op (31-26)	Rs (25-21)	Rt (20-16)	16 bit Address or Immediate (15-0)
---------------	---------------	---------------	---------------------------------------

◆ $R[rt] \leftarrow M[R[rs] + \text{signext}(\text{imm}16)]$

步骤	RTL描述	执行部件	备注
取指令	$IR \leftarrow M[PC]$ $PC \leftarrow PC + 4$	存储器 ALU	指令寄存器IR用于保存读取的指令
读寄存器	$A \leftarrow R[IR[25:21]]$	寄存器堆	寄存器A保存从读取的数据R[rs]
计算地址	$ALUOut \leftarrow A + \text{signext}(IR[15:0])$	ALU	ALUOut保存计算得到的内存地址
访问存储器	$DR \leftarrow M[ALUOut]$	存储器	增加一个寄存器保存数据: DR
写寄存器	$R[IR[20:16]] \leftarrow DR$	寄存器堆	完成寄存器数据写入

4.1 MIPS多周期数据通路设计

3. Sw指令多周期分析

Op (31-26)	Rs (25-21)	Rt (20-16)	16 bit Address or Immediate (15-0)
---------------	---------------	---------------	---------------------------------------

◆ $M[R[rs] + \text{signext}(\text{imm}16)] \leftarrow R[rt]$

步骤	RTL描述	执行部件	备注
取指令	$IR \leftarrow M[PC]$ $PC \leftarrow PC + 4$	存储器 ALU	指令寄存器IR用于保存读取的指令
读寄存器	$A \leftarrow R[IR[25:21]]$	寄存器堆	寄存器A保存从读取的数据R[rs]
计算地址	$ALUOut \leftarrow A +$ $\text{signext}(IR[15:0])$	ALU	ALUOut保存计算得到的内存地址
访问存储器	$M[ALUOut] \leftarrow R[IR[20:16]]$	存储器	把数据R[rt]写入存储器

4.1 MIPS多周期数据通路设计

4. Beq指令多周期分析

Op (31-26)	Rs (25-21)	Rt (20-16)	16 bit Address or Immediate (15-0)
---------------	---------------	---------------	---------------------------------------

- ◆ If (($R[rs] - R[rt]$) == 0) then $PC \leftarrow PC + 4 + \text{signext}(\text{imm16}) \ll 2$
else $PC \leftarrow PC + 4$
- ◆ 问题: ALU 要完成 $PC + 4$, $PC + \text{signext}(\text{imm16}) \ll 2$, $R[rs] - R[rt]$ 三次运算操作, 需要在三个不同时钟周期内实现, 如何合理安排?

步骤	RTL描述	执行部件	备注
取指令	$IR \leftarrow M[PC]$ $PC \leftarrow PC + 4$	存储器 ALU	第一周期后 $PC+4$ 完成, PC 为新值
读寄存器 / 计算转向地址	$A \leftarrow R[IR[25:21]]$ $B \leftarrow R[IR[20:16]]$ $ALUout \leftarrow PC + \text{Signext}[IR[15:0]] \ll 2$	寄存器堆 ALU	ALU 计算转向目标地址, 保存于 $ALUOut$
完成转移	if ($A - B == 0$) then $PC \leftarrow ALUOut$	ALU	ALU 做减法, 结果不写入 ALU, PC 修改是一条件操作

4.1 MIPS多周期数据通路设计

❖ 指令数据通路综合（R型指令、Lw、Sw、Beq）

步骤	R型指令	Lw指令	Sw指令	Beq指令
取指令	$IR \leftarrow M[PC]$ $PC \leftarrow PC + 4$ <div>ALU完成加法 结果送PC</div>			
读寄存器 /译码	$A \leftarrow R[IR[25:21]]$ $B \leftarrow R[IR[20:16]]$ $ALUOut \leftarrow PC + Signext[IR[15:0]] < 2$			
计算	$ALUOut \leftarrow A \text{ op } B$	$ALUOut \leftarrow A + Signext(IR[15:0])$		If (A - B == 0) then $PC \leftarrow ALUOut$
R型完成/ 访问内存	$R[IR[15:11]] \leftarrow ALUOut$	$DR \leftarrow M[ALUOut]$	$M[ALUOut] \leftarrow B$	
写寄存器		$R[IR[20:16]] \leftarrow DR$		

4.1 MIPS多周期数据通路设计

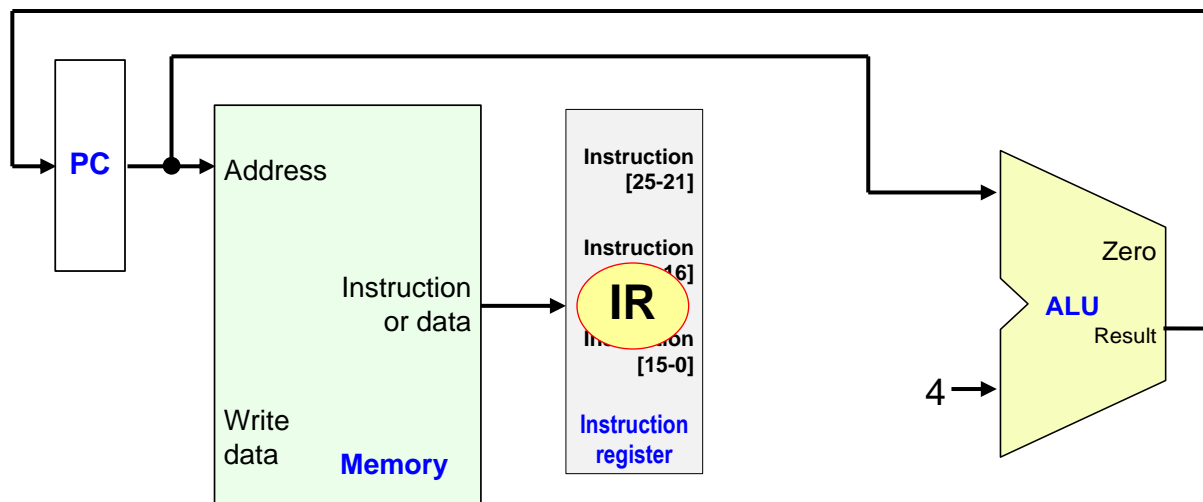
❖ 指令数据通路综合（R型指令、Lw、Sw、Beq）

步骤	R型指令	Lw指令	Sw指令	Beq指令
取指令	$IR \leftarrow M[PC]$ $PC \leftarrow PC + 4$			P0
读寄存器 /译码	$A \leftarrow R[IR[25:21]]$ $B \leftarrow R[IR[20:16]]$ $ALUOut \leftarrow PC + \text{Signext}[IR[15:0]] \ll 2$			P1
计算	$ALUOut \leftarrow A \text{ op } B$	$ALUOut \leftarrow A + \text{Signext}(IR[15:0])$		If $(A-B==0)$ then $PC \leftarrow ALUOut$
R型完成/ 访问内存	$R[IR[15:11]] \leftarrow ALUOut$	$DR \leftarrow M[ALUOut]$	$M[ALUOut] \leftarrow B$	
写寄存器		$R[IR[20:16]] \leftarrow DR$		

4.1 MIPS多周期数据通路设计

1. 多周期数据通路构建（P0）

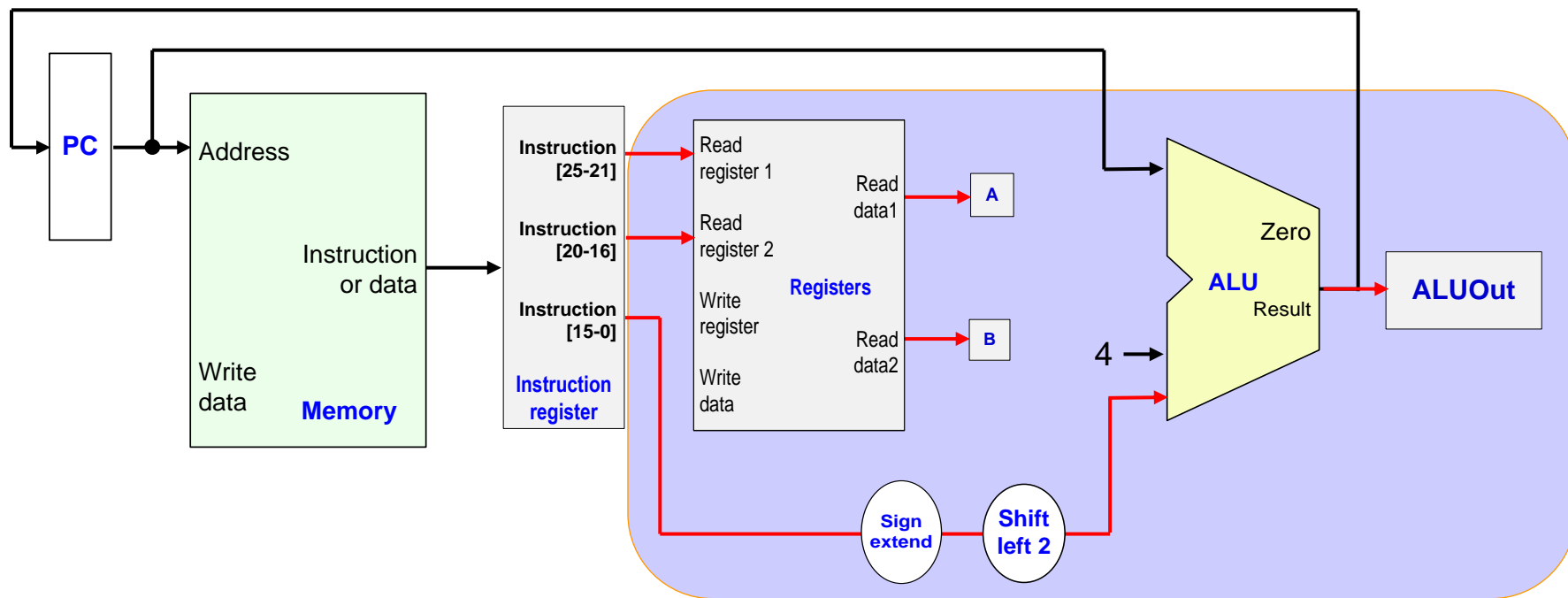
- ◆ $IR \leftarrow M[PC]$
- ◆ $PC \leftarrow PC + 4$



4.1 MIPS多周期数据通路设计

2. 多周期数据通路构建 (P1)

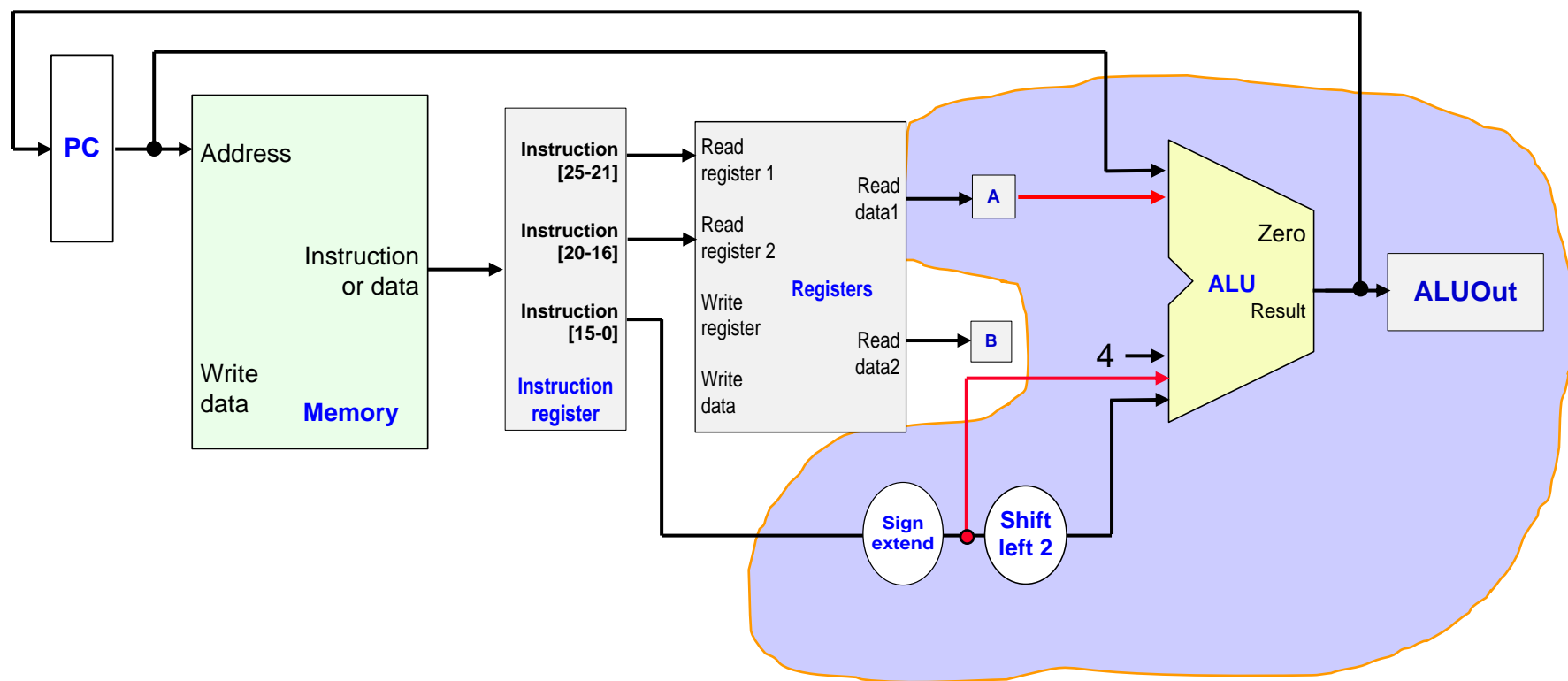
- ◆ $A \leftarrow R[IR[25:21]], B \leftarrow R[IR[20:16]]$
- ◆ $ALUOut \leftarrow PC + Signext[IR[15:0]] \ll 2$



4.1 MIPS多周期数据通路设计

3. 多周期数据通路构建 (P2)

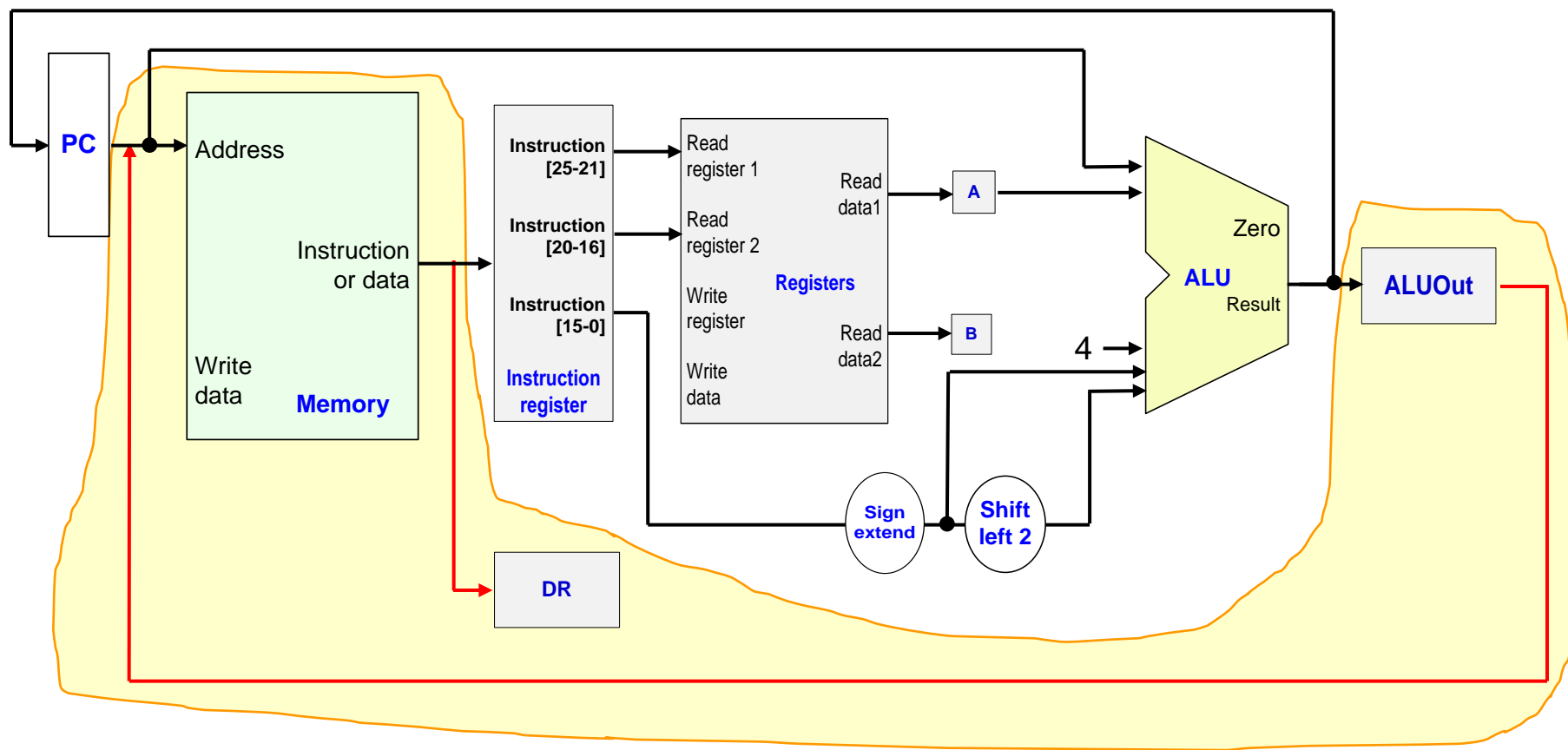
◆ $ALUOut \leftarrow A + Signext[IR[15:0]]$



4.1 MIPS多周期数据通路设计

4. 多周期数据通路构建 (P3)

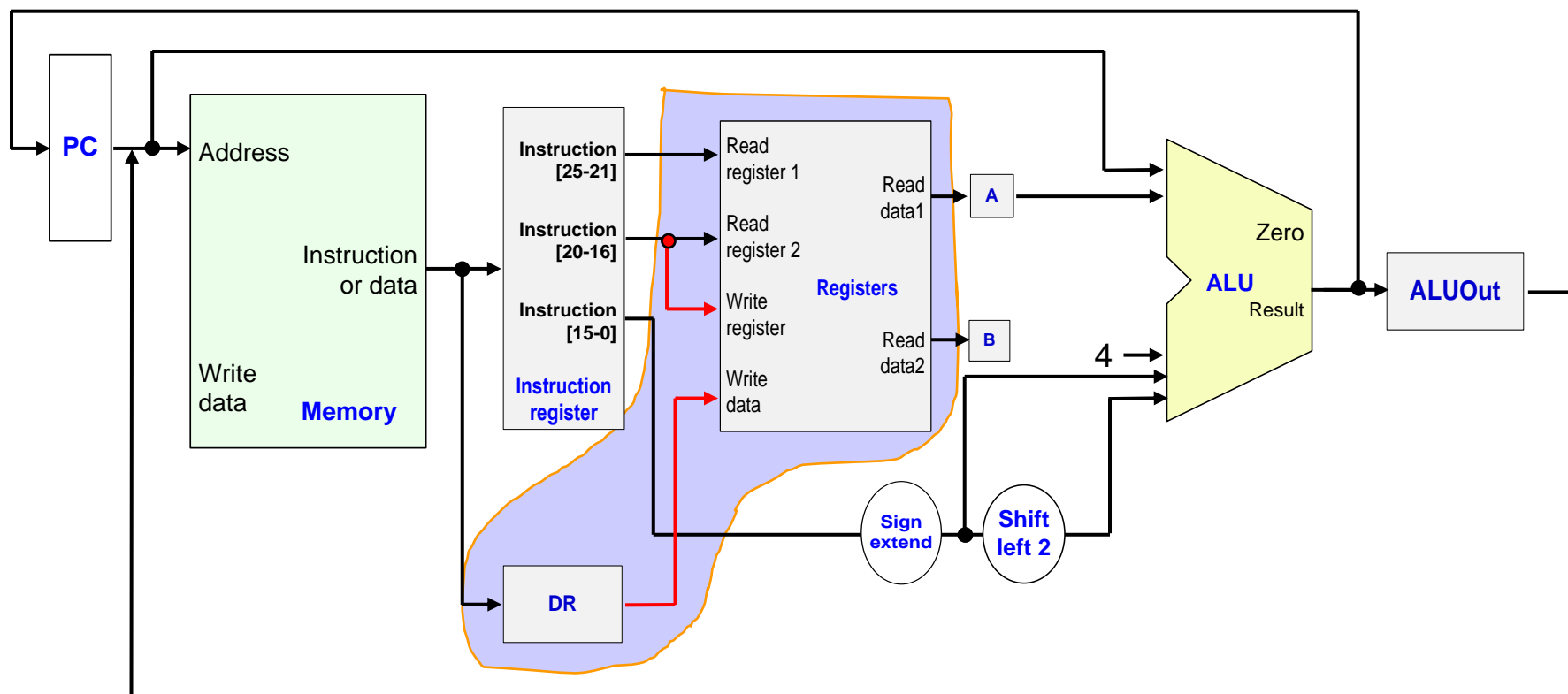
◆ $DR \leftarrow M[ALUOut]$



4.1 MIPS多周期数据通路设计

5. 多周期数据通路构建 (P4)

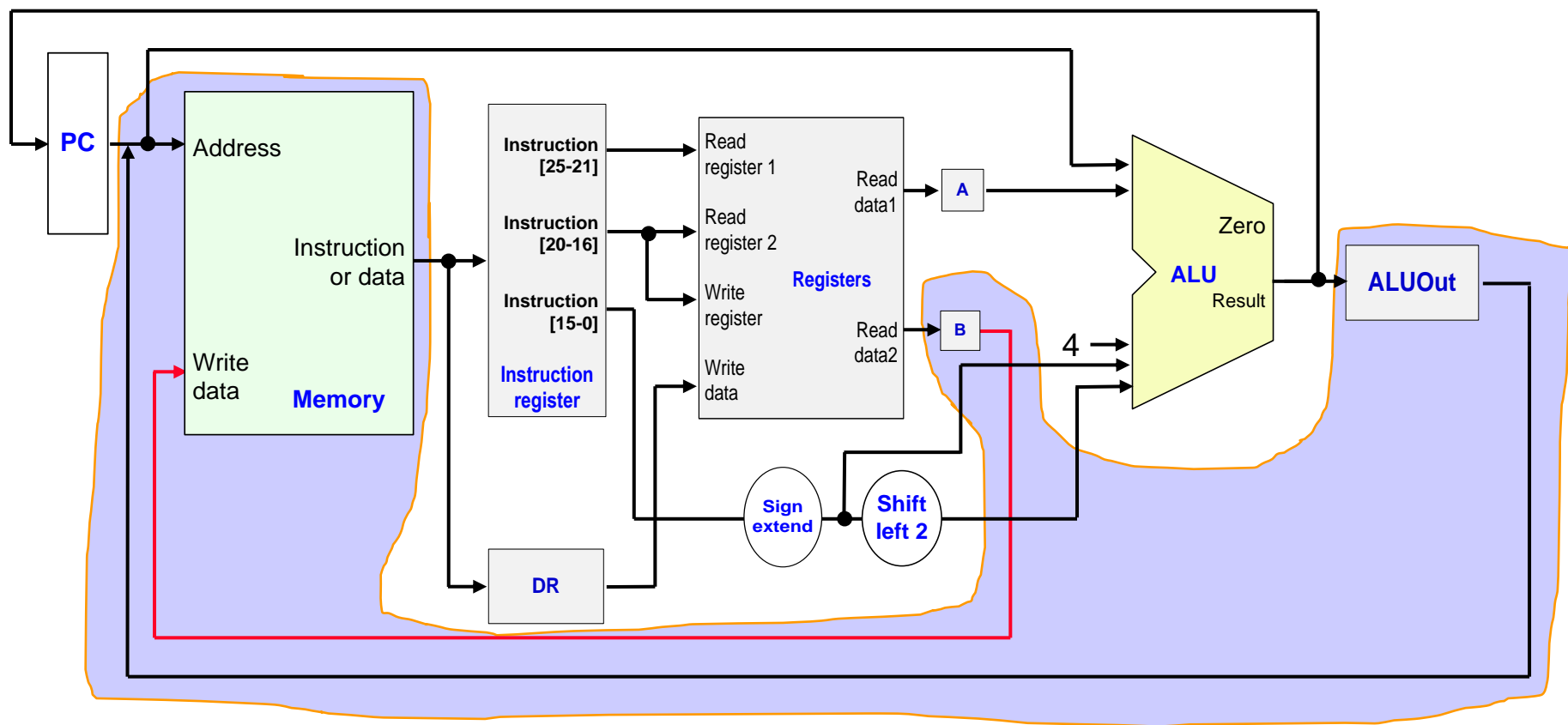
◆ $R[IR[20:16]] \leftarrow DR$



4.1 MIPS多周期数据通路设计

6. 多周期数据通路构建 (P5)

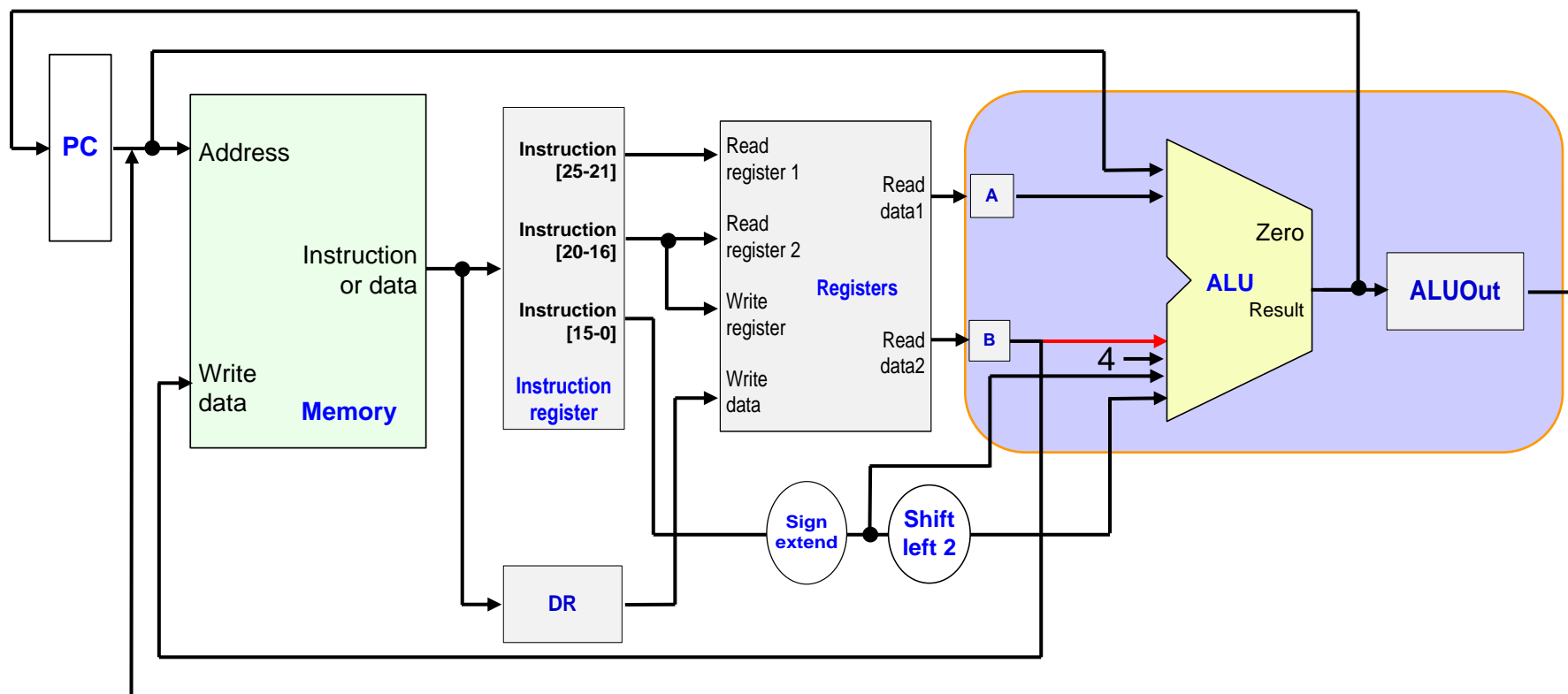
◆ $M[ALUOut] \leftarrow B$



4.1 MIPS多周期数据通路设计

7. 多周期数据通路构建 (P6)

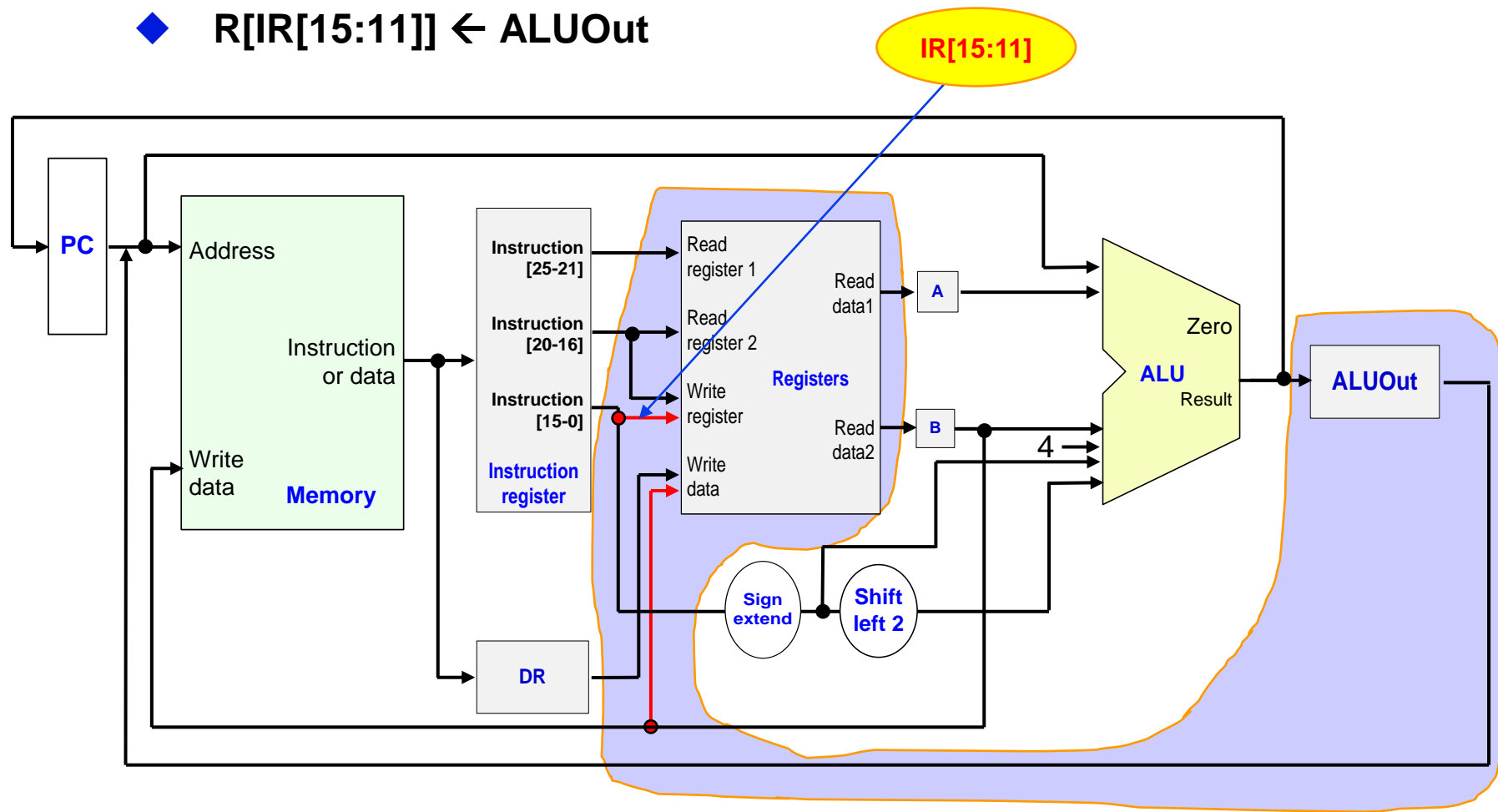
◆ $AUOut \leftarrow A \text{ op } B$



4.1 MIPS多周期数据通路设计

8. 多周期数据通路构建 (P7)

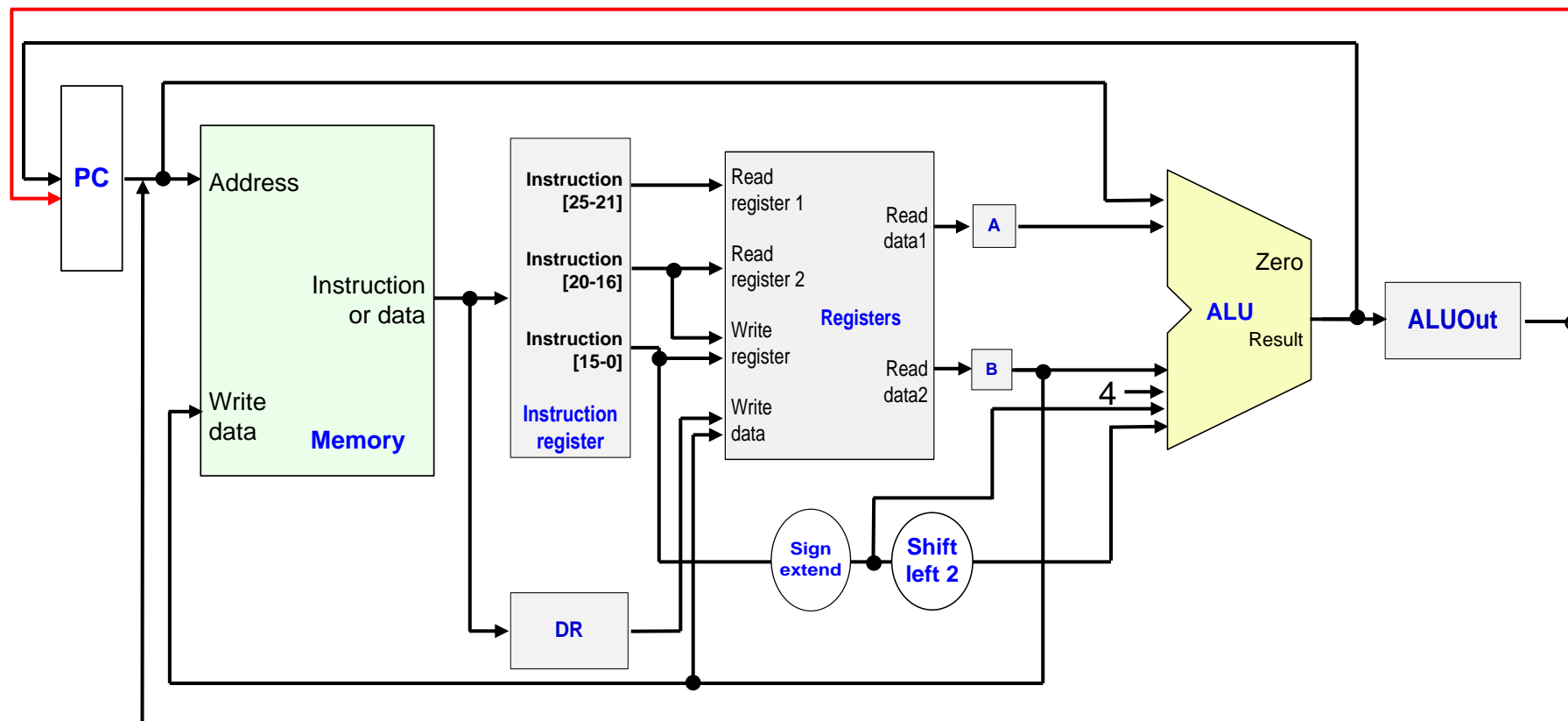
◆ $R[IR[15:11]] \leftarrow ALUOut$



4.1 MIPS多周期数据通路设计

9. 多周期数据通路构建 (P8)

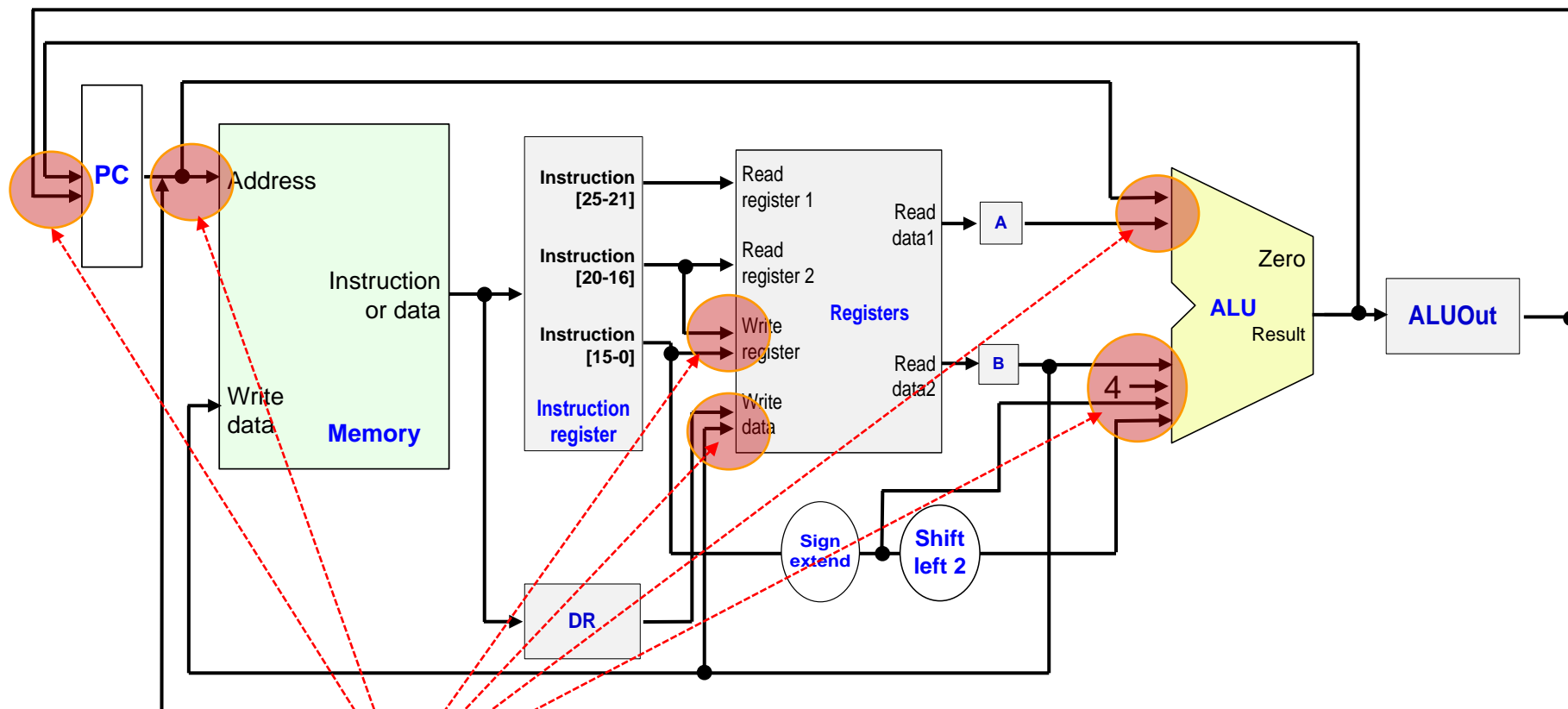
◆ If (A-B==0) then $PC \leftarrow ALUOut$



4.1 MIPS多周期数据通路设计

10. 多周期数据通路构建

- ◆ 存在多个输入源的，需要增加多路选择器MUX

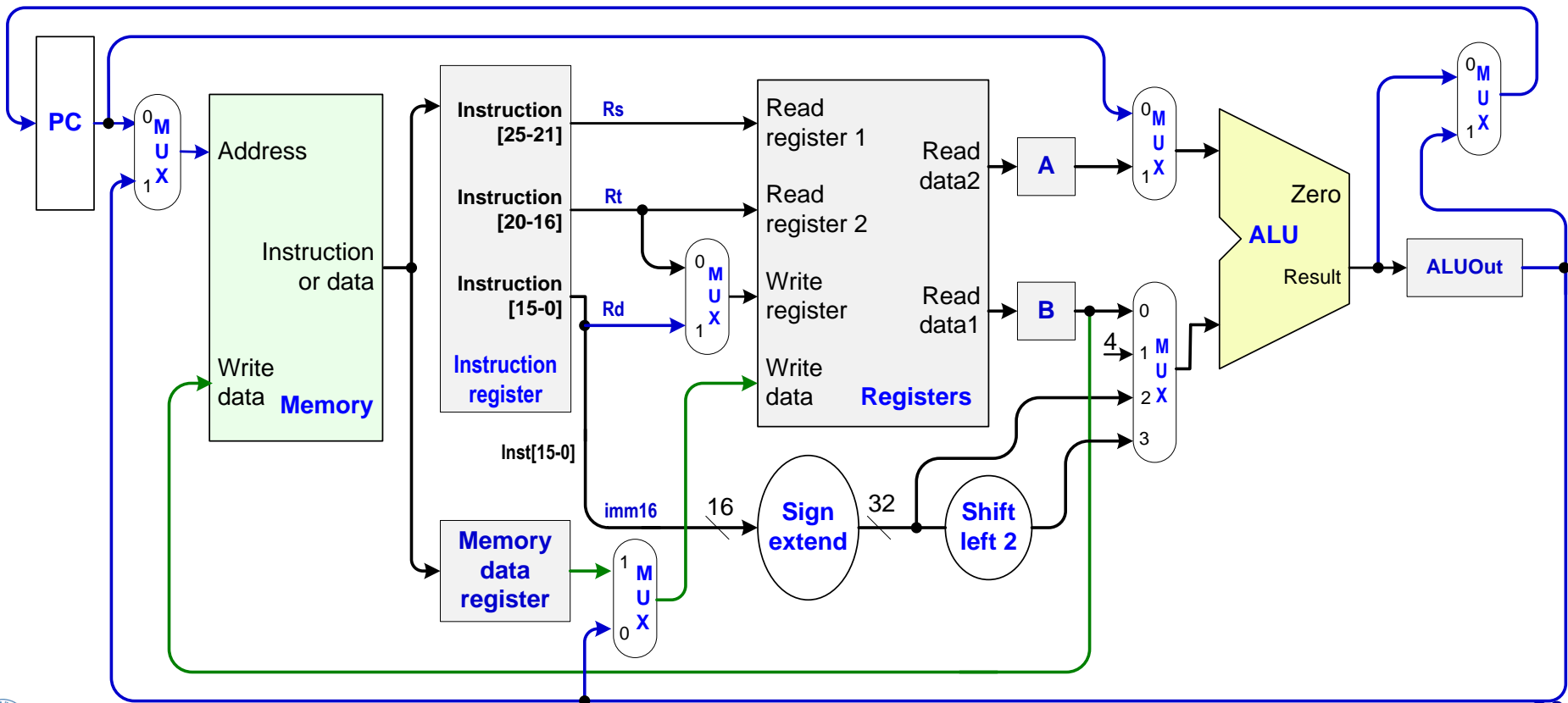


加入多路选择MUX

4.1 MIPS多周期数据通路实现

❖ MIPS的多周期数据通路

- **PC**: 输入端多路选择 (2选1)
- **存储器**: 地址输入端多路选择 (2选1)
- **ALU**: 输入A端多路选择 (2选1), 输入B端多路选择 (4选1)
- **寄存器堆**: 写地址输入端 (2选1), 写数据输入端 (2选1)



4.1 MIPS多周期数据通路设计

■ 增加J指令（转移）通路

Op
(31-26)

26 bit Address (for Jump Instruction)
(25-0)

➤ $PC \leftarrow PC + 4 [31:28] \parallel add26 < < 2$

步骤	R型指令	Lw指令	Sw指令	Beq指令	J指令
取指令	$IR \leftarrow M[PC]$ $PC \leftarrow PC + 4$				
读寄存器/译码	$A \leftarrow R[IR[25:21]]$ $B \leftarrow R[IR[20:16]]$ $ALUOut \leftarrow PC + Signext[IR[15:0]] < < 2$				
计算	$ALUOut \leftarrow A \text{ op } B$	$ALUOut \leftarrow A + Signext(IR[15:0])$		If (A-B==0) then $PC \leftarrow ALUout$	$PC \leftarrow PC[31:28] \parallel IR[25:0] < < 2$
R型完成/访问内存	$R[IR[15:11]] \leftarrow ALUOut$	$DR \leftarrow M[ALUOut]$	$M[ALUOut] \leftarrow B$		
写寄存器		$R[IR[20:16]] \leftarrow DR$			

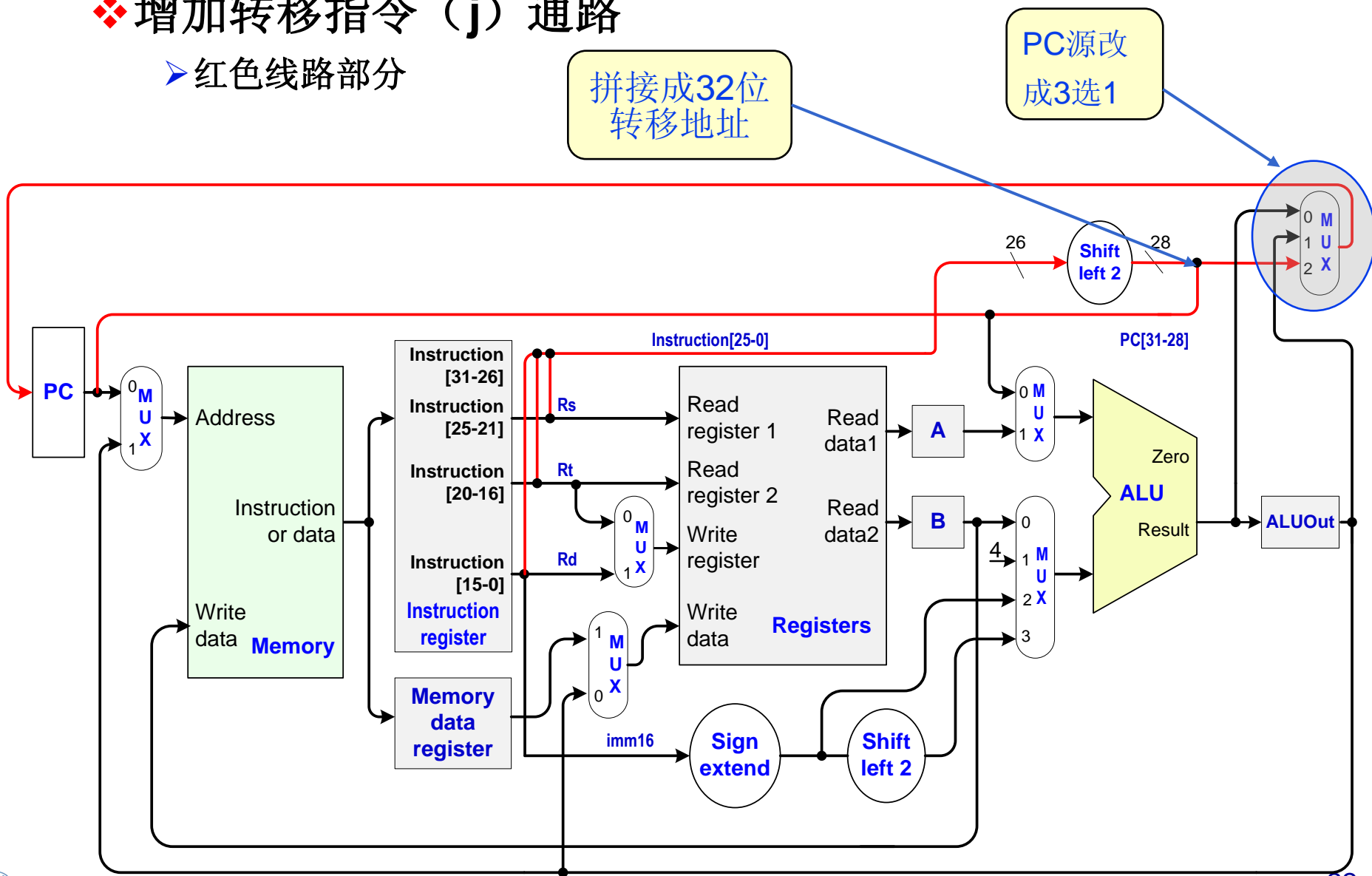
4.1 MIPS多周期数据通路实现

❖ 增加转移指令 (j) 通路

➤ 红色线路部分

拼接成32位
转移地址

PC源改
成3选1

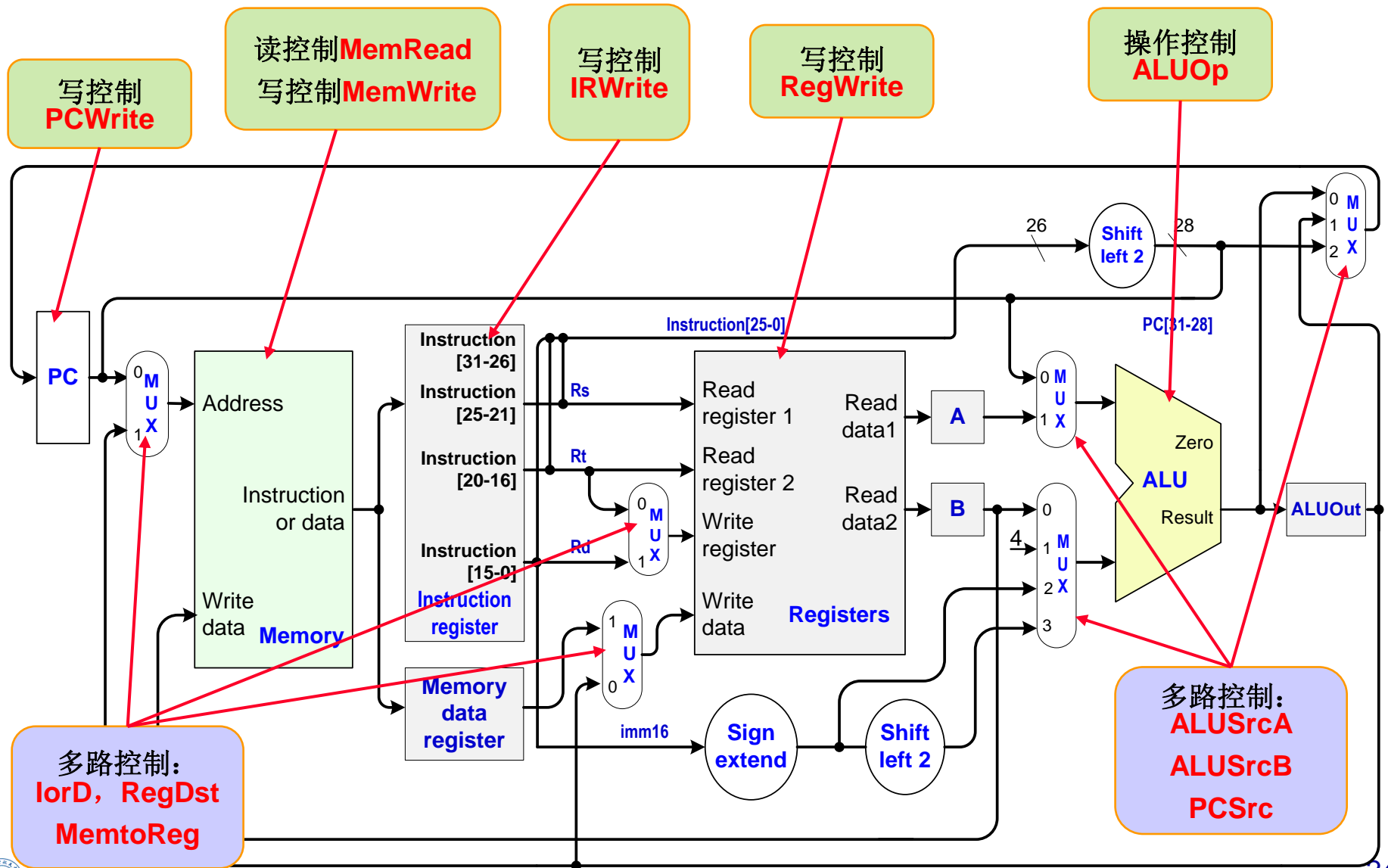


第六讲 MIPS处理器设计

- 一. 处理器设计概述
- 二. MIPS模型机
- 三. MIPS单周期处理器设计
 - 1. 单周期数据通路设计
 - 2. 单周期控制器设计
 - 3. 单周期性能分析
- 四. MIPS多周期处理器设计**
 - 1. 多周期数据通路设计
 - 2. 多周期控制器设计**
 - 3. 多周期性能分析**

4.2 多周期控制器设计

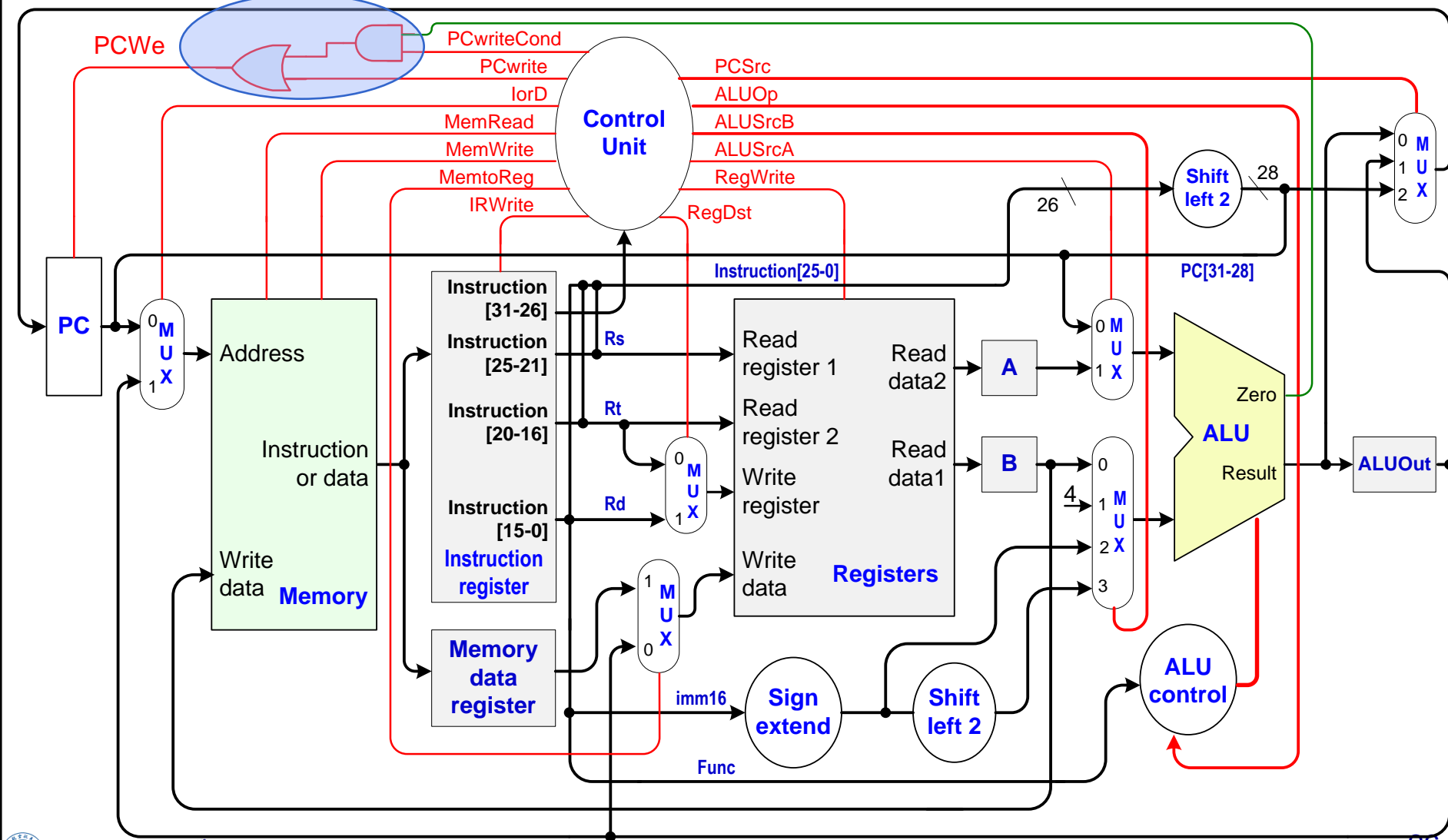
❖ 数据通路控制信号设计





4.2 多周期控制器设计——完整数据通路与控制信号

PCWe = PCWrite | PCWriteCond & Zero



4.2 多周期控制器设计

多周期通路控制信号

控制信号	失效时作用	有效时作用
RegDst	寄存器堆写入端地址来选择Rt字段	寄存器堆写入端地址选择Rd字段
RegWrite	无	把数据写入寄存器堆中对应寄存器
ALUSrcA	ALU输入A端选择PC	ALU输入A端选择寄存器A
MemRead	无	存储器读数据（输出）
MemWrite	无	存储器写数据（输入）
MemtoReg	寄存器堆写入端数据选择ALUOut	寄存器堆写入端数据选择DR
IorD	存储器地址输入选择PC	存储器地址输入选择ALUOut
IRWrite	无	存储器输出（指令）写入IR
PCWrite	无	PC写入，PC输入源由PCSrc选择
PCWriteCond	无	如ALU的Zero端输出有效，则PC写入，输入源由PCSrc选择（Beq指令）

4.2 多周期控制器设计

多周期通路控制信号（续）

控制信号	取值 (二进制)	作用
ALUOp	00	ALU执行加法
	01	ALU执行减法
	10	ALU操作由Func字段 (IR[5:0]) 决定
ALUSrcB	00	ALU输入端B数据源选择寄存器B
	01	ALU输入端B数据源选择常数4
	10	ALU输入端B选择符号扩展输出 (Signext(imm16))
	11	ALU输入端B选择移位器输出 (Signext(imm16)<<2)
PCSrc	00	PC输入源选择ALU输出 (取指阶段, PC+4)
	01	PC输入源选择寄存器 ALUOut (Beq指令)
	10	PC输入源选择转移地址 PC[31:28] // IR[25:0]<<2 (j指令)

4.2 多周期控制器设计

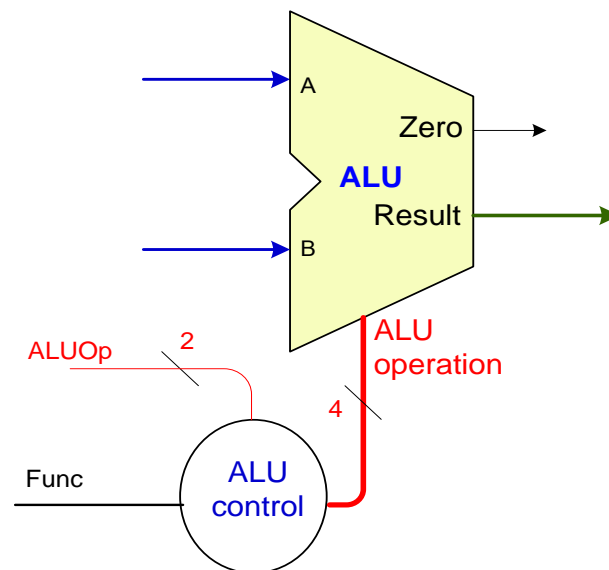
❖ ALU控制单元

➤ 输入:

- 指令的**Func**字段（指令**5:0**位）
- 由主控单元生成的 **ALUOp**

➤ **ALUOp**指明**ALU**的运算类型

- **00**: 访存指令所需的加法
- **01**: **beq**指令所需的减法
- **10**: **R**型指令功能码字段决定



指令	Func字段	ALUop	ALU运算类型	ALU operation
Lw	XXXXXX	00	加	0010
Sw	XXXXXX	00	加	0010
Beq	XXXXXX	01	减	0110
Add	100 000	10	加	0010
Sub	100 010	10	减	0110
And	100 100	10	与	0000
Or	100 101	10	或	0001

ALU控制单元真值表

4.2 多周期控制器设计

多周期数据通路汇总

时钟周期	R型指令	Lw指令	Sw指令	Beq指令	J指令
TC1	$IR \leftarrow M[PC]$ $PC \leftarrow PC + 4$				
TC2	$A \leftarrow R[IR[25:21]], B \leftarrow R[IR[20:16]]$ $ALUOut \leftarrow PC + \text{Signext}[IR[15:0]] \ll 2$				
TC3	$ALUOut \leftarrow A \text{ op } B$	$ALUOut \leftarrow A + \text{Signext}(IR[15:0])$	$\text{If } (A-B==0) \text{ then } PC \leftarrow ALUOut$	$PC \leftarrow PC[31:28] \parallel IR[25:0] \ll 2$	
TC4	$R[IR[15:11]] \leftarrow ALUOut$	$DR \leftarrow M[ALUOut]$	$M[ALUOut] \leftarrow B$		
TC5		$R[IR[20:16]] \leftarrow DR$			

4.2 多周期控制器设计

1. 数据通路控制信号状态（P0，所有指令TC1）

- ◆ $IR \leftarrow M[PC]$
- ◆ $PC \leftarrow PC + 4$

通路 部件	存储器 地址 MUX	存储器	IR	ALU-A 端MUX	ALU-B 端MUX	ALU	PC输入 端MUX	PC
操作	选择PC	读出	写入	选择PC	常数4	加	选择ALU	写入
控制 信号 状态	lorD=0	MemRead	IRWrite	ALUSrcA = 0	ALUSrcB = 01	ALUOp = 00	PCSrc = 00	PCWrite

状态**ST0**:

- MemRead有效
- IRWrite有效
- PCWrite有效
- lorD = 0
- ALUSrcA = 0
- ALUSrcB = 01
- ALUOp = 00
- PCSrc = 00

4.2 多周期控制器设计

2. 数据通路控制信号状态（**P1**，所有指令**TC2**）

- ◆ $A \leftarrow R[IR[25:21]]$, $B \leftarrow R[IR[20:16]]$
- ◆ $ALUOut \leftarrow PC + \text{Signext}[IR[15:0]] \ll 2$

通路部件	寄存器堆	ALU-A端MUX	ALU-B端MUX	ALU
操作	无	选择PC	选择移位器	加
控制信号状态	无	$ALUSrcA = 0$	$ALUSrcB = 11$	$ALUOp = 00$

状态**ST1**:

- $ALUSrcA = 0$
- $ALUSrcB = 11$
- $ALUOp = 00$

4.2 多周期控制器设计

3. 数据通路控制信号状态（P2, Lw/Sw指令TC3）

◆ $ALUOut \leftarrow A + \text{Signext}(IR[15:0])$

通路 部件	ALU-A端MUX	ALU-B端MUX	ALU
操作	选择寄存器A	选择 符号扩展	加
控制 信号 状态	$ALUSrcA = 1$	$ALUSrcB = 10$	$ALUOp = 00$

状态**ST2**:

- $ALUSrcA = 1$
- $ALUSrcB = 10$
- $ALUOp = 00$

4.2 多周期控制器设计

4. 数据通路控制信号状态（P3, Lw指令TC4）

◆ $DR \leftarrow M[ALUOut]$

通路 部件	存储器地址端 MUX	存储器	数据寄存器DR
操作	选择ALUOut	读出	写入
控制 信号 状态	lorD = 1	MemRead	无

状态ST3:

- lorD = 1
- MemRead有效

4.2 多周期控制器设计

5. 数据通路控制信号状态（P4, Lw指令TC5）

◆ $R[IR[20:16]] \leftarrow DR$

通路 部件	寄存器堆写 地址MUX	寄存器堆写 数据MUX	寄存器堆
操作	选择Rt	选择DR	写入
控制 信号 状态	RegDst = 0	MemtoReg = 1	RegWrite

状态ST4:

- RegDst = 0
- MemtoReg = 1
- RegWrite有效

4.2 多周期控制器设计

6. 数据通路控制信号状态（P5, Sw指令TC4）

◆ $M[ALUOut] \leftarrow B$

通路 部件	存储器 地址MUX	存储器	
操作	选择ALUOut	写入	
控制 信号 状态	lorD = 1	MemWrite	

状态ST5:

- lorD = 1
- MemWrite有效

4.2 多周期控制器设计

7. 数据通路控制信号状态（P6，R型指令TC3）

◆ $ALUOut \leftarrow A \text{ op } B$

通路 部件	ALU-A端MUX	ALU-B端MUX	ALU
操作	选择 寄存器A	选择 寄存器B	根据Func字段 运算
控制 信号 状态	ALUSrcA = 1	ALUSrcB = 00	ALUOp = 10

状态ST6:

- ALUSrcA = 1
- ALUSrcB = 00
- ALUOp = 10

4.2 多周期控制器设计

8. 数据通路控制信号状态（P7，R型指令TC4）

◆ $R[IR[15:11]] \leftarrow ALUOut$

通路 部件	寄存器堆写 地址MUX	寄存器堆写 数据MUX	寄存器堆
操作	选择Rd	选择ALUOut	写入
控制 信号 状态	RegDst = 1	MemtoReg = 0	RegWrite

状态ST7:

- RegDst = 1
- MemtoReg = 0
- RegWrite有效

4.2 多周期控制器设计

9. 数据通路控制信号状态（P8, Beq指令TC3）

◆ If ($A - B == 0$) then $PC \leftarrow ALUout$

通路 部件	ALU-A端 MUX	ALU-B端 MUX	ALU	PC输入端 MUX	PC
操作	选择 寄存器A	选择 寄存器B	减	选择 ALUOut	有条件写入
控制 信号 状态	ALUSrcA = 1	ALUSrcB = 00	ALUOp = 01	PCSrc = 01	PCWriteCond

状态**ST8**:

- ALUSrcA = 1
- ALUSrcB = 00
- ALUOp = 01
- PCSrc = 01
- PCWriteCond有效

4.2 多周期控制器设计

10. 数据通路控制信号状态（P9，J指令TC3）

◆ $PC \leftarrow PC[31:28] \parallel IR[25:0] \ll 2$

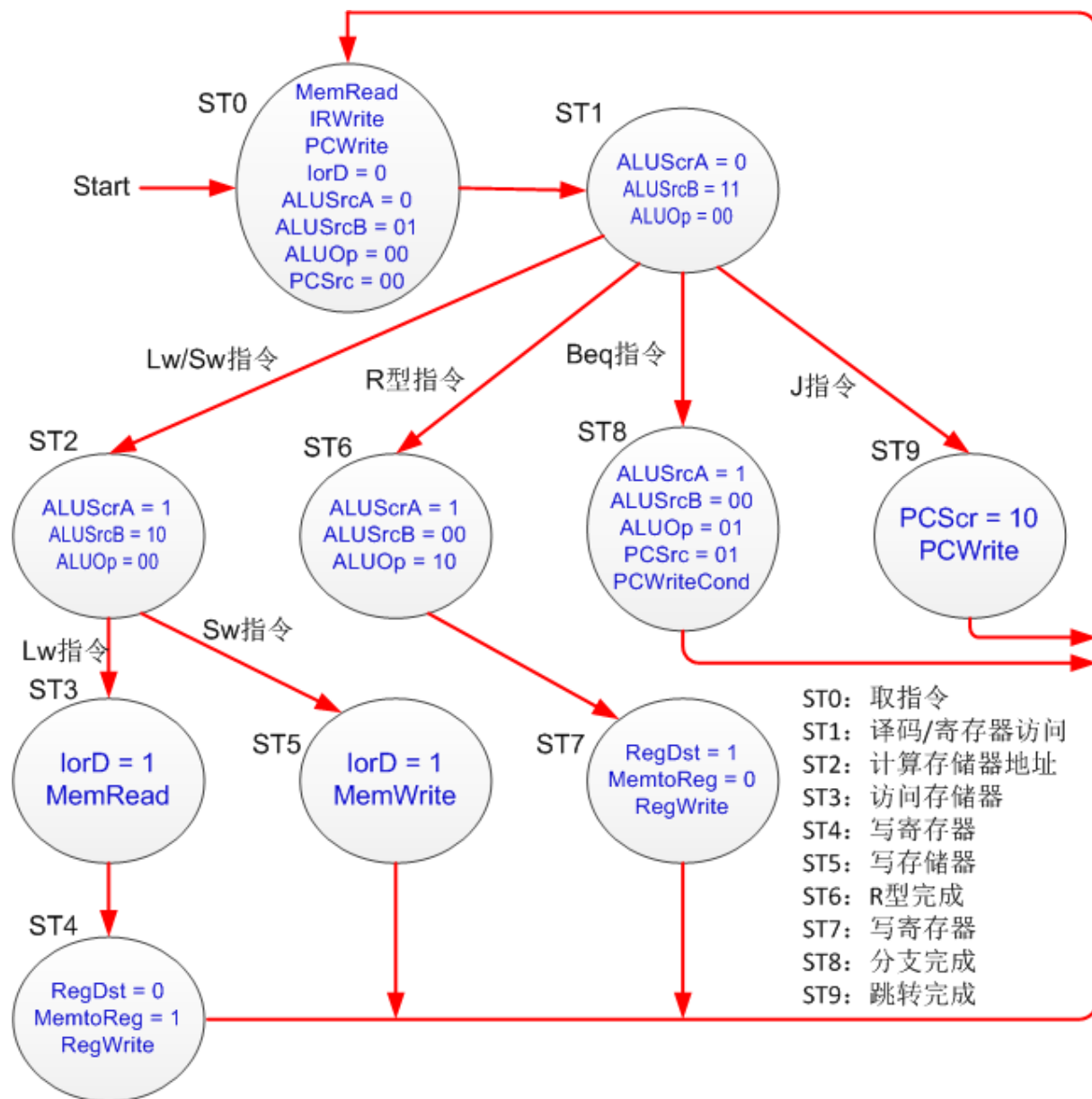
通路 部件	PC输入端MUX	PC	
操作	选择转移地址	写入	
控制 信号 状态	PCSrc = 10	PCWrite	

状态ST9:

- PCSrc = 10
- PCWrite有效

4.2 多周期控制器设计——主控单元

多周期
主控单元
有限状态图



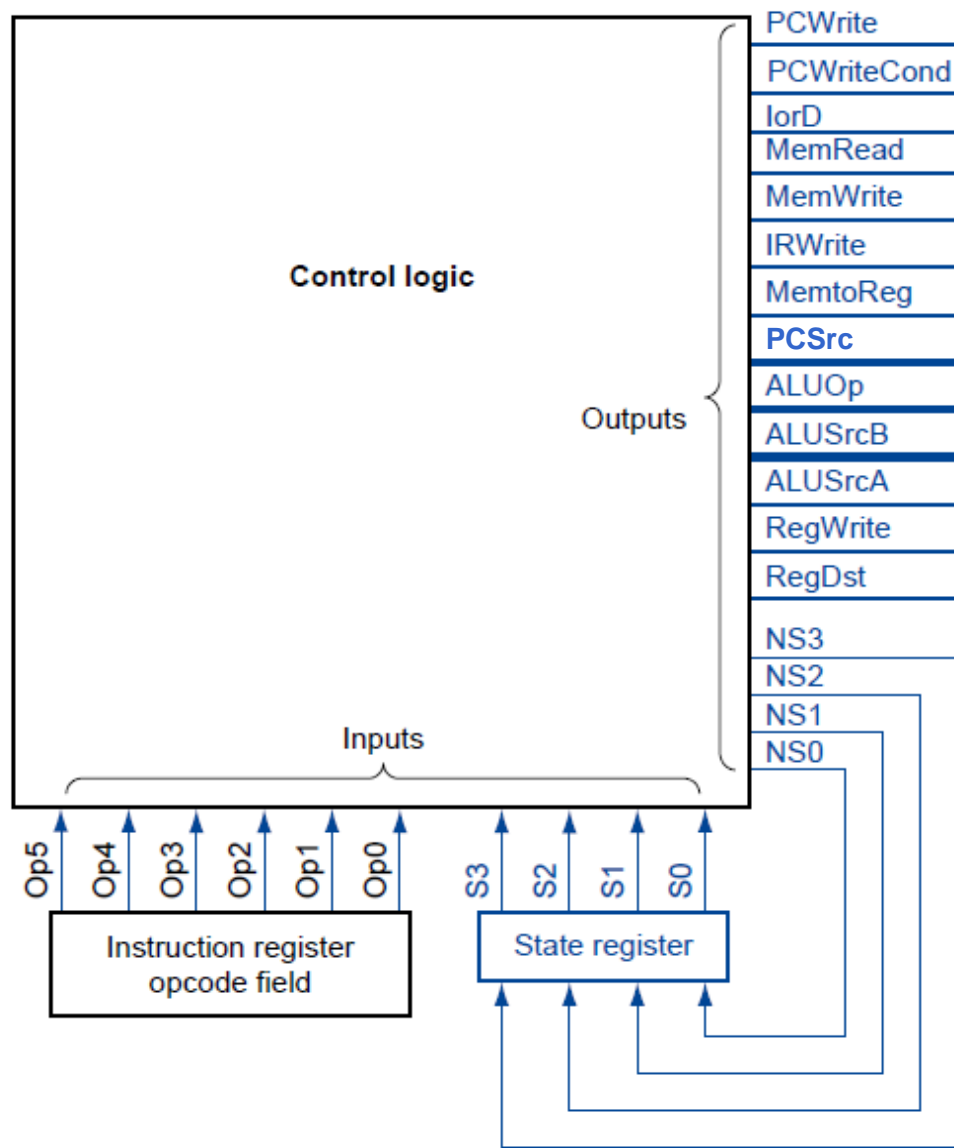
4.2 多周期控制器设计——主控制单元

❖ 有限状态机FSM

- ST0、ST1等10个状态
- 状态编码用4位二进制表示，如 $S_3S_2S_1S_0=0110$ 表示ST6
- 次态用 $NS_3NS_2NS_1NS_0$ 表示

❖ 摩尔型FSM

- 控制信号输出仅取决于当前状态
- 次态由现态和当前输入（IR寄存器中6位操作码）决定



4.2 多周期控制器设计——主控制单元

❖ 控制信号逻辑示例

控制信号仅和当前状态有关

$$PCWrite = ST0 + ST9$$

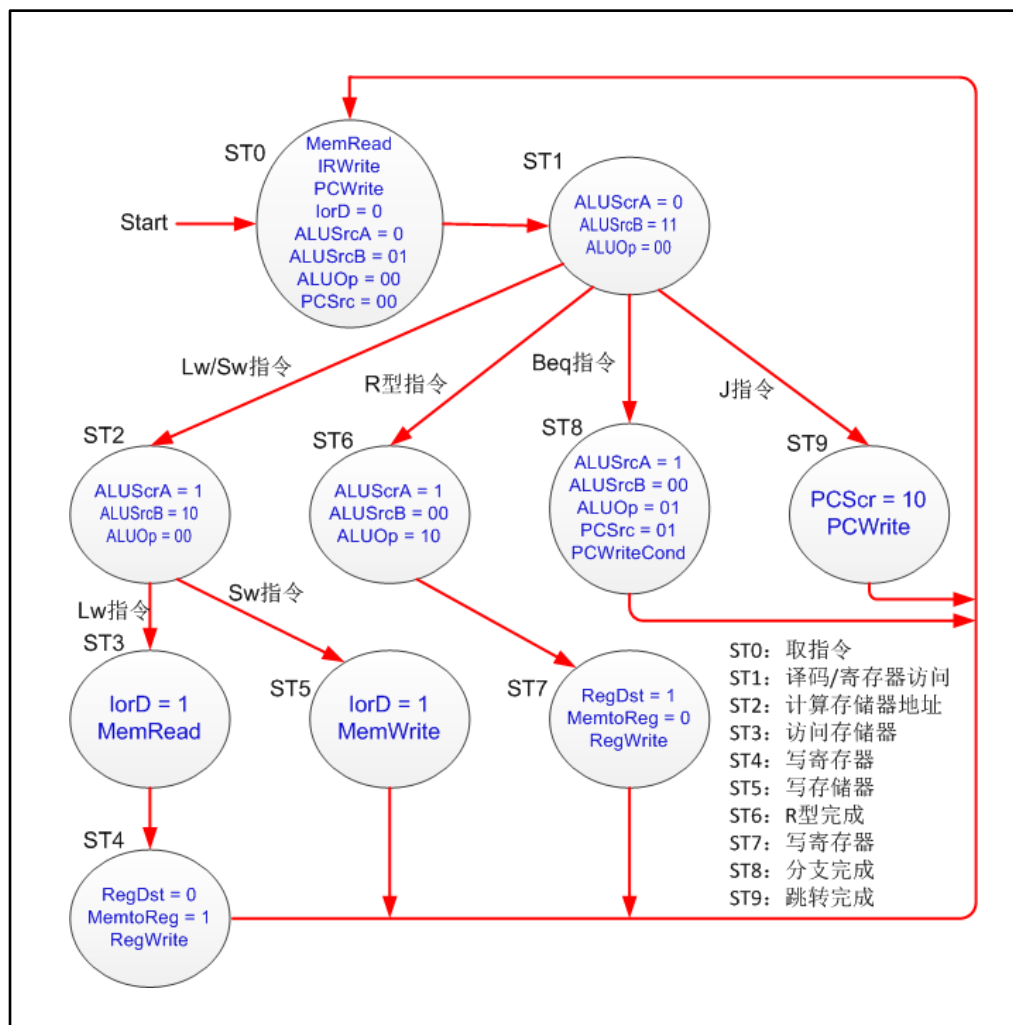
$$= \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + \overline{S_3} \overline{S_2} \overline{S_1} S_0$$

$$ALUSrcA = ST2 + ST6 + ST8$$

$$= \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + \overline{S_3} \overline{S_2} \overline{S_1} S_0 + \overline{S_3} \overline{S_2} S_1 \overline{S_0}$$

$$RegWrite = ST4 + ST7$$

$$= \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + \overline{S_3} \overline{S_2} S_1 S_0$$



4.2 多周期控制器设计——主控制单元

❖ 次态信号逻辑： 次态输出与当前状态和当前输入（Opcode）相关

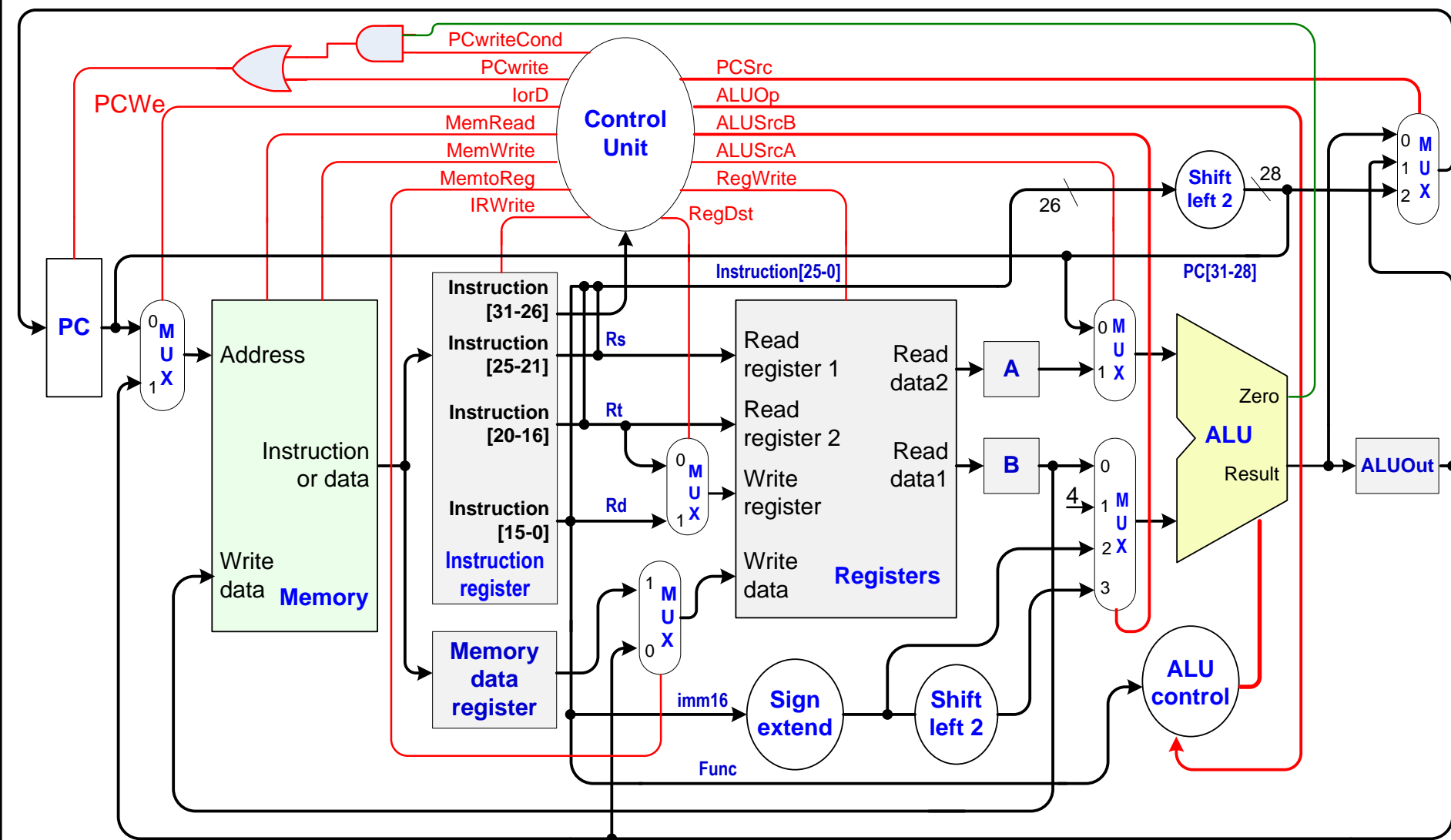
次态信号真值表

当前状态输入 $S_3S_2S_1S_0$	操作码输入 ($Op_5Op_4Op_3Op_2Op_1Op_0$)	次态输出 $NS_3NS_2NS_1NS_0$
0000 (ST0)		0001 (ST1)
0001 (ST1)	100011 (lw指令)	0010 (ST2)
0001 (ST1)	101011 (sw指令)	0010 (ST2)
0001 (ST1)	000000 (R型指令)	0110 (ST6)
0001 (ST1)	000100 (Beq指令)	1000 (ST8)
0001 (ST1)	000010 (J指令)	1001 (ST9)
0010 (ST2)	100011 (lw指令)	0011 (ST3)
0010 (ST2)	101011 (sw指令)	0101 (ST5)
0011 (ST3)		0100 (ST4)
0110 (ST6)		0111 (ST7)
ST4, ST5, ST7, ST8, ST9		0000 (ST0)

$$NS_3 = ST1(Beq + J)$$

$$= \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} (\overline{Op_5} \overline{Op_4} \overline{Op_3} \overline{Op_2} \overline{Op_1} \overline{Op_0} + \overline{Op_5} \overline{Op_4} \overline{Op_3} \overline{Op_2} \overline{Op_1} \overline{Op_0})$$

多周期控制器设计 —— 主控单元 + ALU控制单元



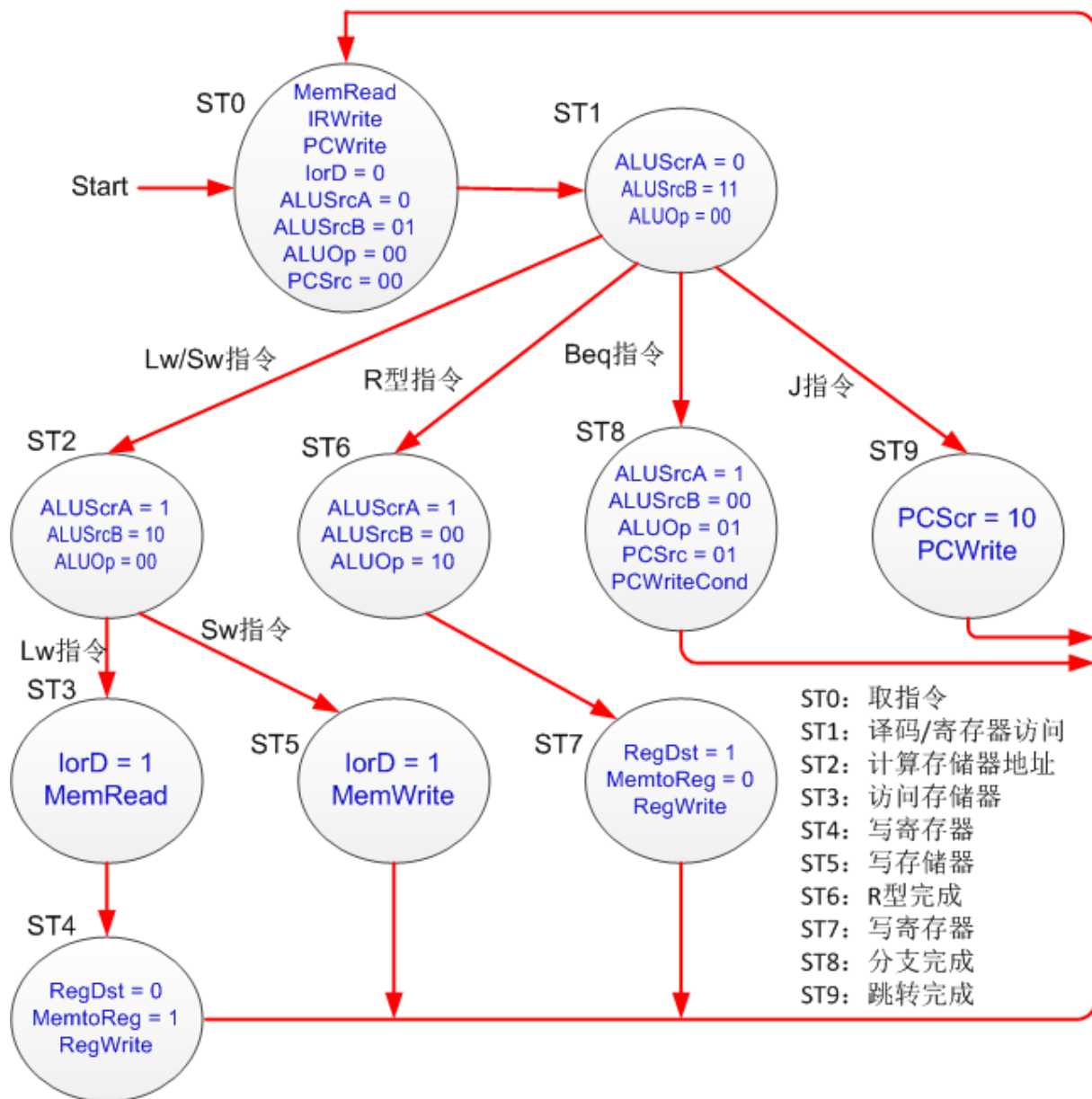
多周期控制器设计 —— 主控单元

多周期数据通路汇总

时钟周期	R型指令	Lw指令	Sw指令	Beq指令	J指令
TC1	$IR \leftarrow M[PC]$ $PC \leftarrow PC + 4$				
TC2	$A \leftarrow R[IR[25:21]], B \leftarrow R[IR[20:16]]$ $ALUOut \leftarrow PC + \text{Signext}[IR[15:0]] \ll 2$				
TC3	$ALUOut \leftarrow A \text{ op } B$	$ALUOut \leftarrow A + \text{Signext}(IR[15:0])$	$\text{If } (A-B==0) \text{ then } PC \leftarrow ALUOut$	$PC \leftarrow PC[31:28] \parallel IR[25:0] \ll 2$	
TC4	$R[IR[15:11]] \leftarrow ALUOut$	$DR \leftarrow M[ALUOut]$	$M[ALUOut] \leftarrow B$		
TC5		$R[IR[20:16]] \leftarrow DR$			

多周期控制器设计 —— 主控单元

多周期
主控单元
有限状态图



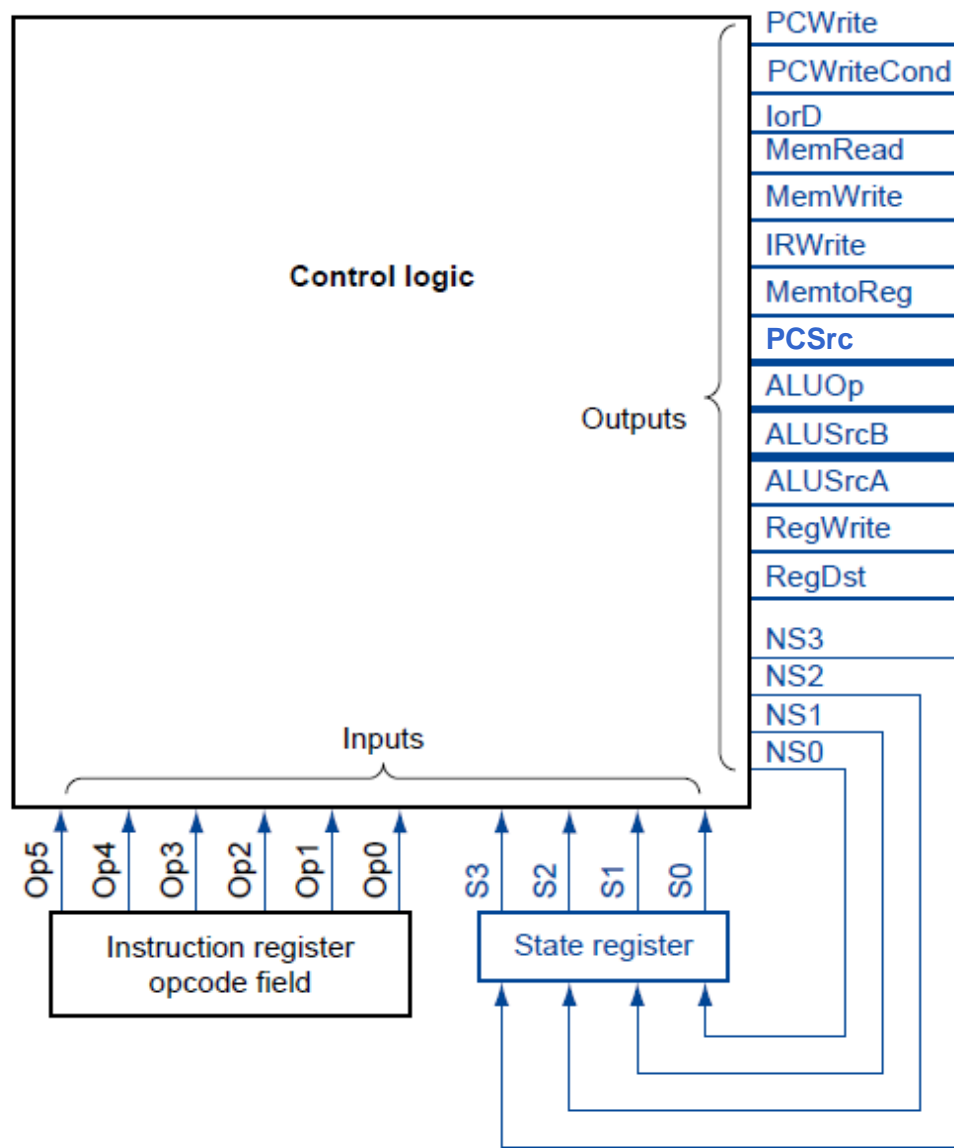
多周期控制器设计 —— 主控制单元

❖ 有限状态机FSM

- ST0、ST1等10个状态
- 状态编码用4位二进制表示，如 $S_3S_2S_1S_0=0110$ 表示ST6
- 次态用 $NS_3NS_2NS_1NS_0$ 表示

❖ 摩尔型FSM

- 控制信号输出仅取决于当前状态
- 次态由现态和当前输入（IR寄存器中6位操作码）决定



第六讲 MIPS处理器设计

- 一. 处理器设计概述
- 二. MIPS模型机
- 三. MIPS单周期处理器设计
 - 1. 单周期数据通路设计
 - 2. 单周期控制器设计
 - 3. 单周期性能分析
- 四. MIPS多周期处理器设计**
 - 1. 多周期数据通路设计
 - 2. 多周期控制器设计
 - 3. 多周期性能分析**

4.3 多周期性能分析

❖ 假设主要功能单元的操作时间

- 存储器 : 200ps
- ALU : 100ps
- 寄存器堆: 50ps
- 多路复用器、控制单元、PC、符号扩展单元、线路没有延迟

各类指令执行时间

步骤	R型指令	Lw指令	Sw指令	Beq指令	J指令	执行时间
取指令	$IR \leftarrow M[PC], PC \leftarrow PC + 4$					200ps
读寄存器/ 译码	$A \leftarrow R[IR[25:21]], B \leftarrow R[IR[20:16]]$ $ALUOut \leftarrow PC + Signext[IR[15:0]] \ll 2$					100ps
计算	$ALUOut \leftarrow A \text{ op } B$	$ALUOut \leftarrow A + Signext(IR[15:0])$		If $(A-B==0)$ then $PC \leftarrow ALUout$	$PC \leftarrow PC[31:28] \parallel IR[25:0] \ll 2$	100ps
R型完成/ 访问内存	$R[IR[15:11]] \leftarrow ALUOut$	$DR \leftarrow M[ALUOut]$	$M[ALUOut] \leftarrow B$			200ps
写寄存器		$R[IR[20:16]] \leftarrow DR$				50ps

4.2 多周期控制器设计

❖ 时钟周期

➤ 时钟周期取各步骤中最长的时间: **200ps**

各类指令执行时间

时钟周期	R型指令	Lw指令	Sw指令	Beq指令	J指令	周期时间
TC1	$IR \leftarrow M[PC], PC \leftarrow PC + 4$					200ps
TC2	$A \leftarrow R[IR[25:21]], B \leftarrow R[IR[20:16]]$ $ALUOut \leftarrow PC + Signext[IR[15:0]] \ll 2$					200ps
TC3	$ALUOut \leftarrow A \text{ op } B$	$ALUOut \leftarrow A + Signext(IR[15:0])$		If $(A-B==0)$ then $PC \leftarrow ALUout$	$PC \leftarrow PC[31:28] \parallel IR[25:0] \ll 2$	200ps
TC4	$R[IR[15:11]] \leftarrow ALUOut$	$DR \leftarrow M[ALUOut]$	$M[ALUOut] \leftarrow B$			200ps
TC5		$R[IR[20:16]] \leftarrow DR$				200ps

4.3 多周期性能分析

❖ 各型指令所需的时钟周期数和时间

- R型指令 : 800ps
- lw指令 : 1000ps
- sw指令 : 800ps
- beq指令 : 600ps
- j指令 : 600ps

❖ 假设指令在程序中出现的频率

- lw指令 : 25%
- sw指令 : 10%
- R型指令: 45%
- beq指令: 15%
- j指令 : 5%

❖ 则一条指令的平均CPI

- $5 \times 25\% + 4 \times 10\% + 4 \times 45\% + 3 \times 15\% + 3 \times 5\% = 4.05$

❖ 一条指令的平均执行时间:

- $1000 \times 25\% + 800 \times 10\% + 800 \times 45\% + 600 \times 15\% + 600 \times 5\% = 810\text{ps}$