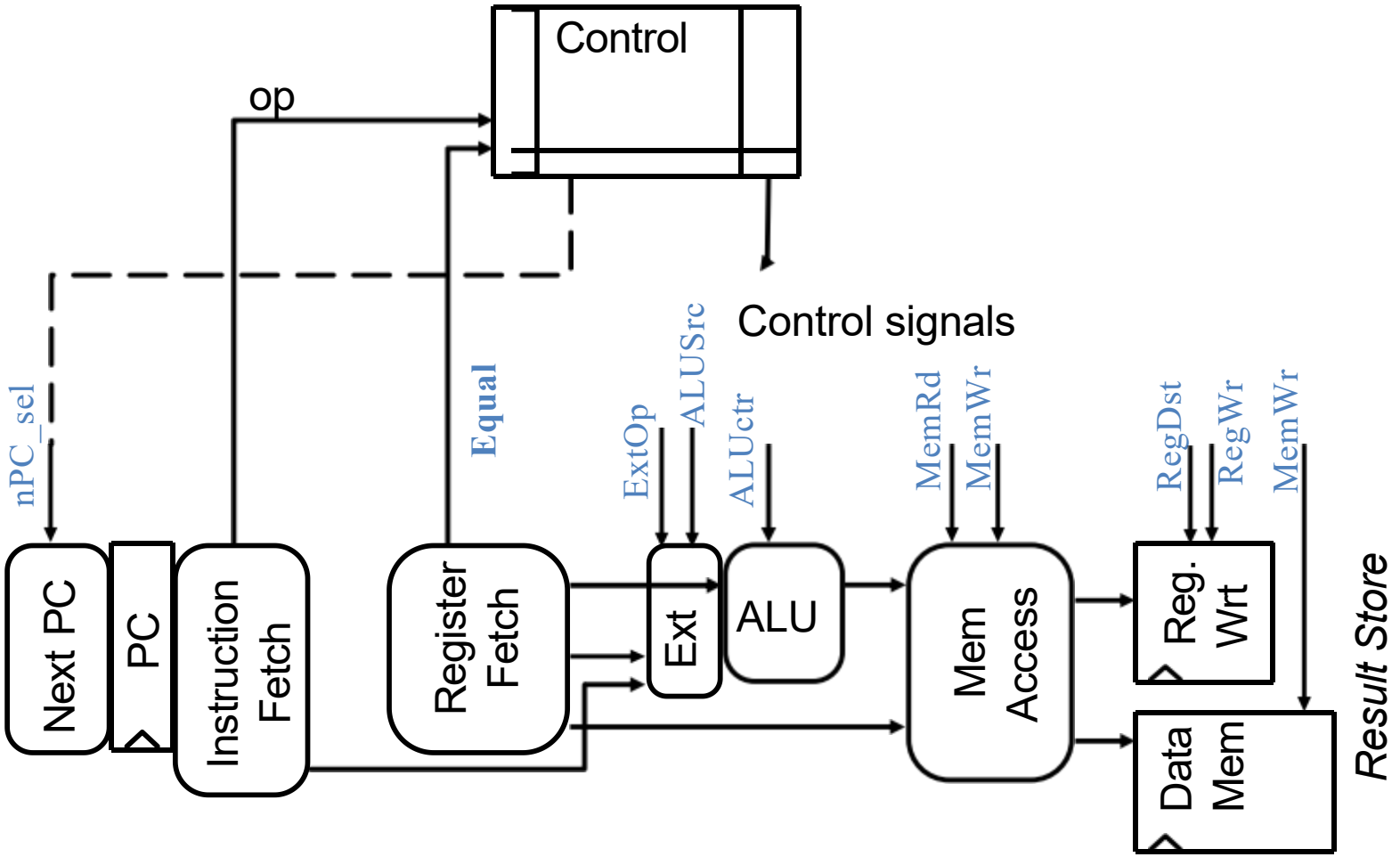
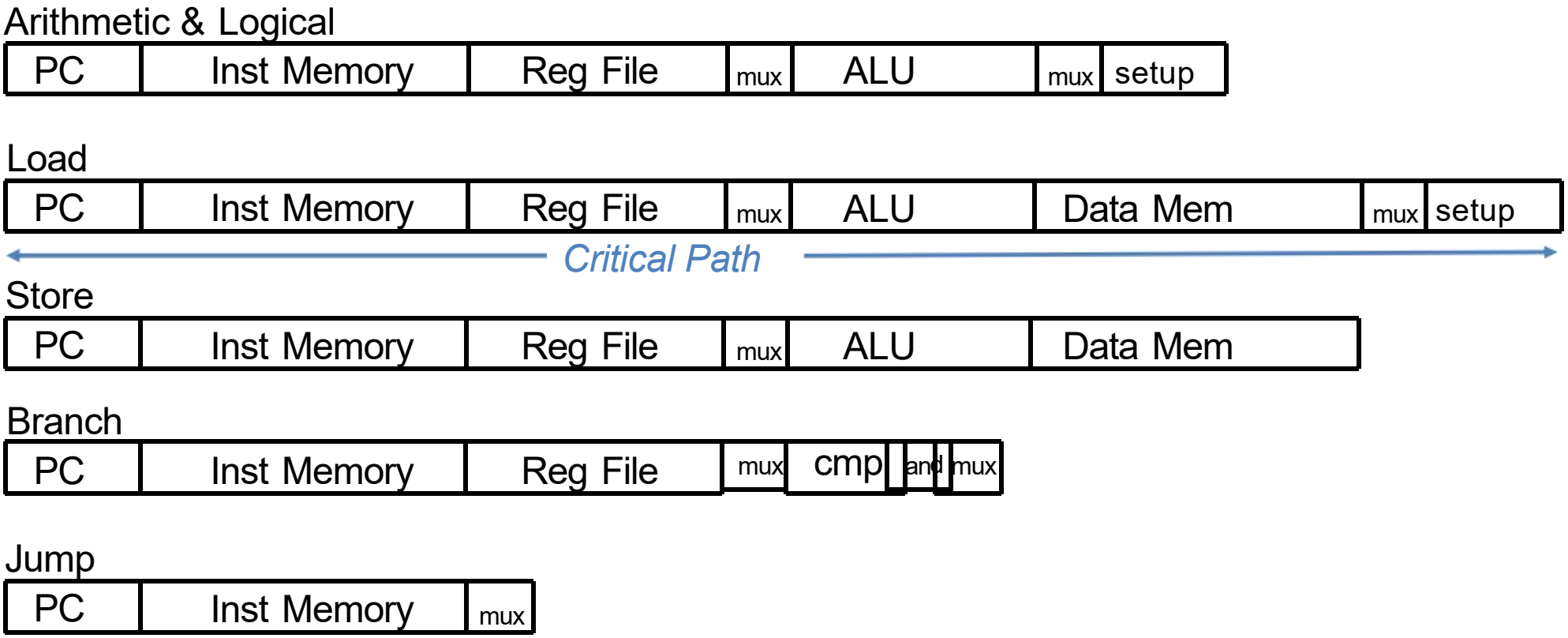
处理器架构

多周期处理器的微体系结构

单周期处理器的抽象视图



What's wrong with our CPI=1 processor?



. 长周期时间

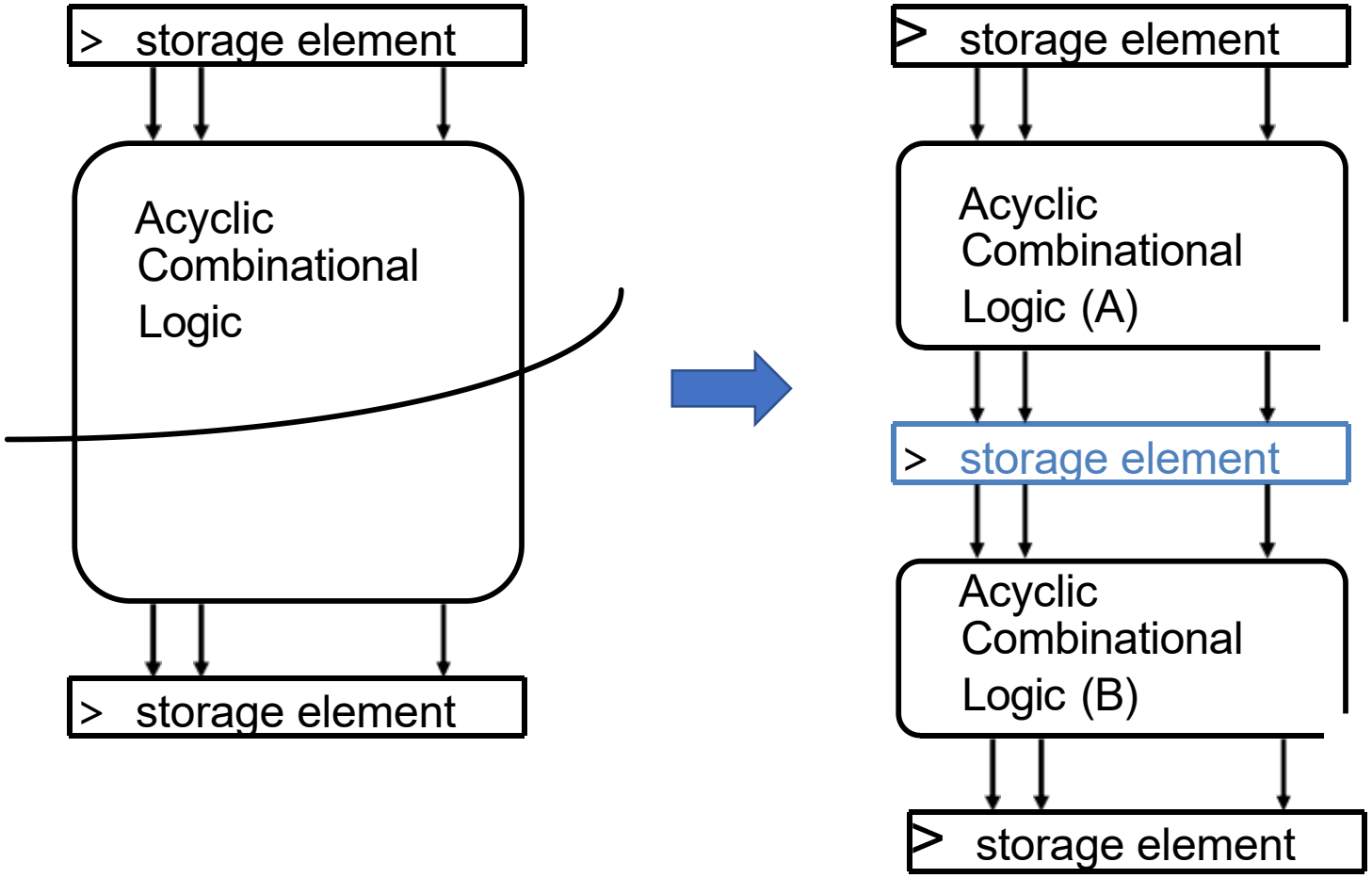
. 所有指令所花费的时间与最慢的指令相同

. 现实的存储器并不像我们理想的存储器

——不能总是在一个（短）周期内完成工作

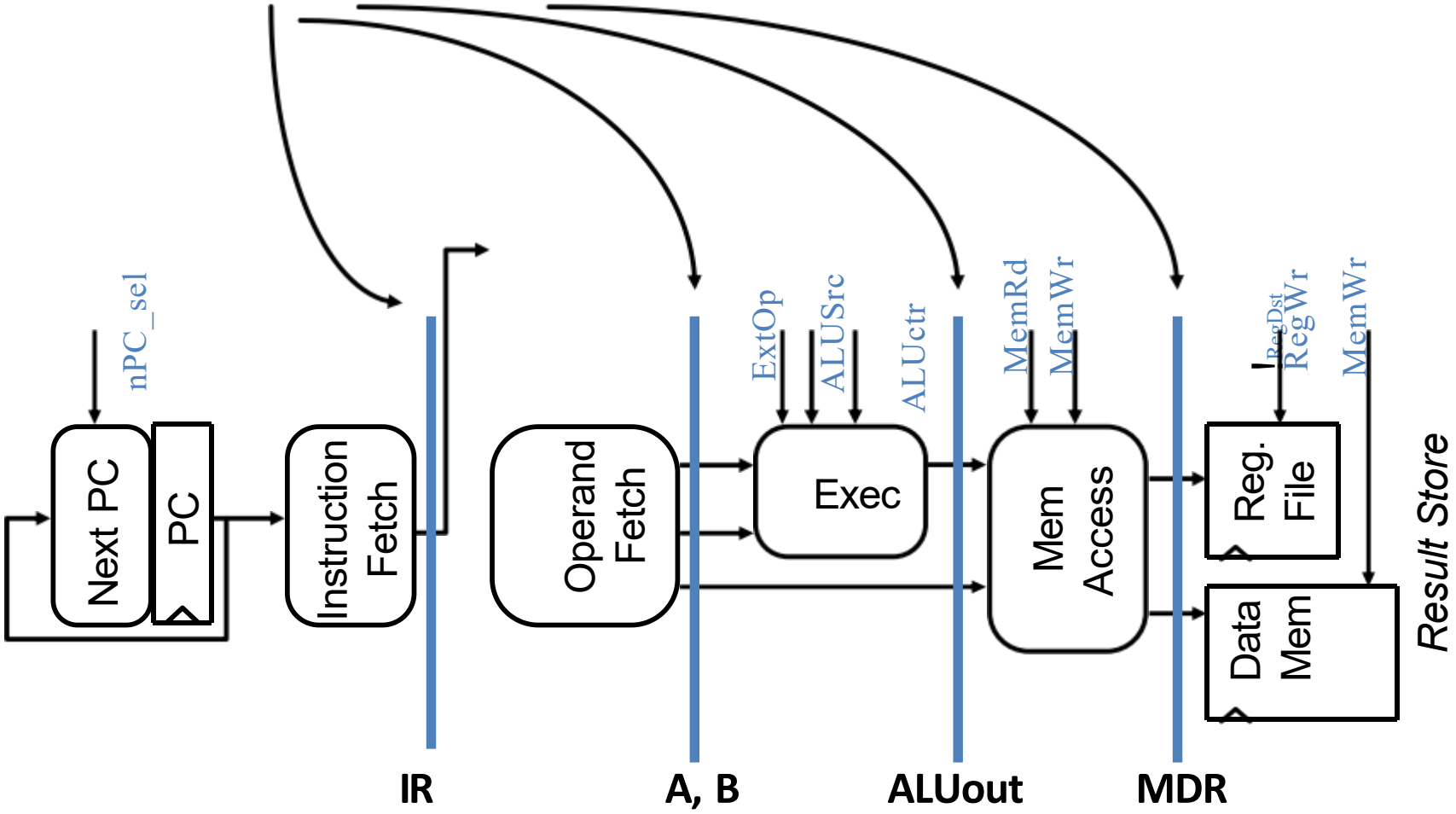
减少周期时间

- 切断组合逻辑相关的连接并添加寄存器
- 在两个快周期做重复性工作，而不是慢周期中完成同样的工作。



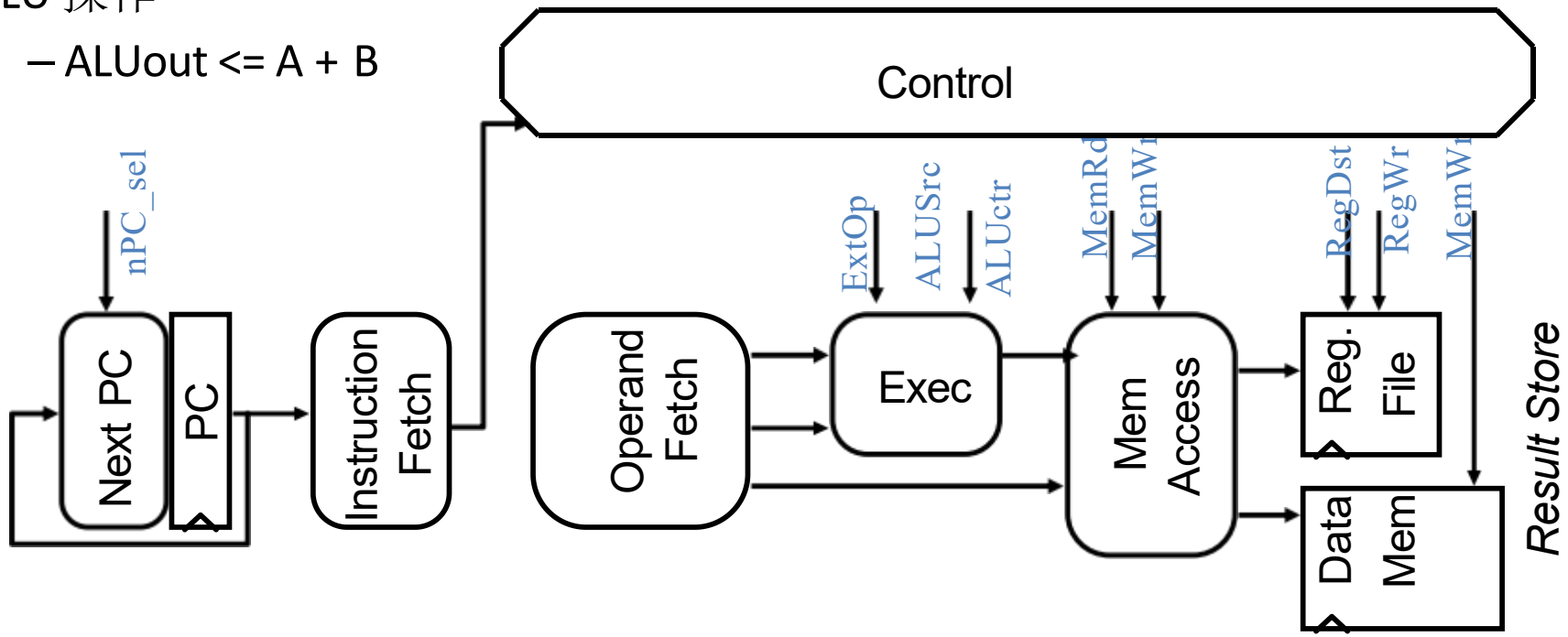
对CPI=1 的数据路径进行分区

- 在最小步长之间添加寄存器

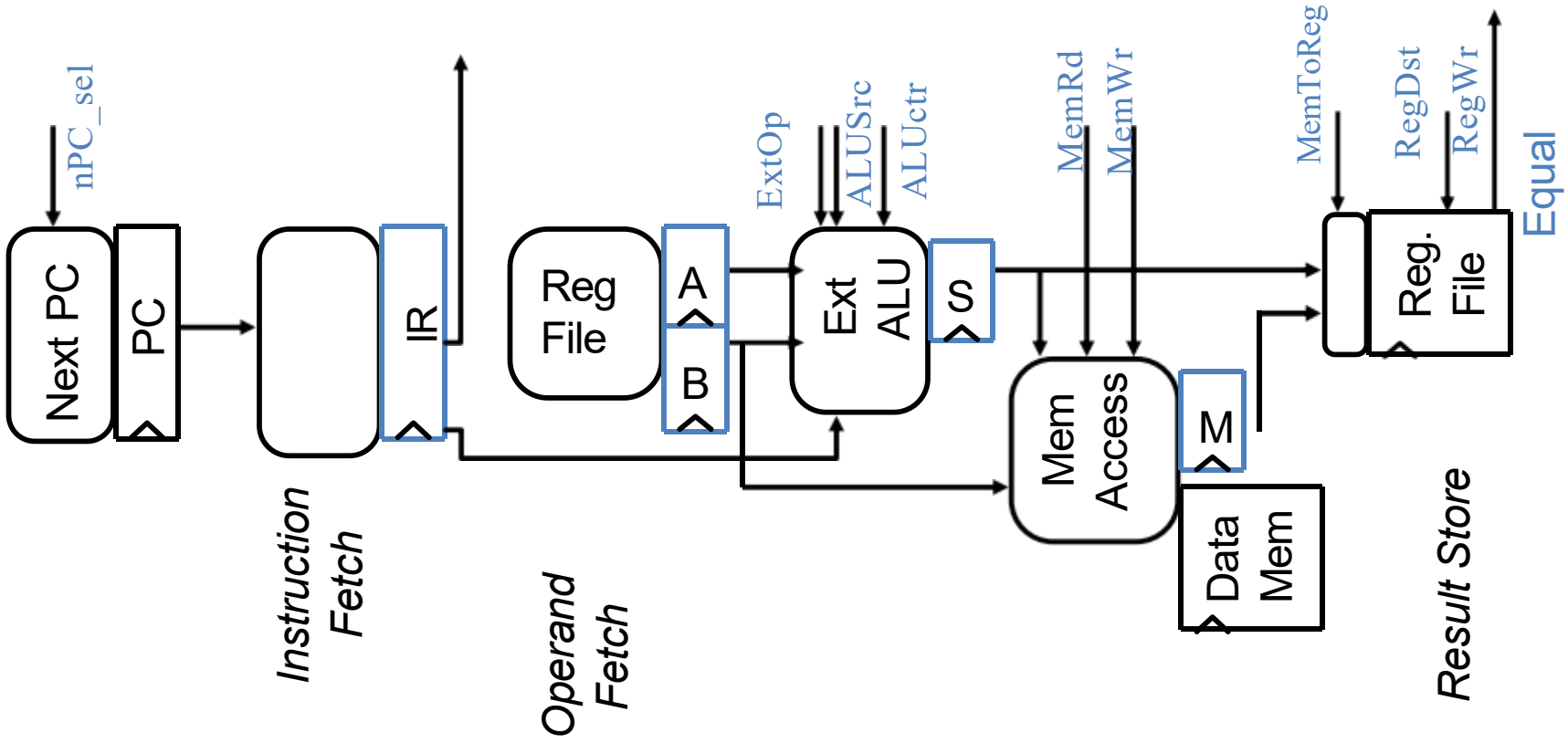


周期时间的基本限制

- 下一个地址逻辑
  - PC <= branch ? PC + 1 or PC + offset
- 取指
  - InstructionReg <= Mem[PC]
- 寄存器访问
  - A <= R[rn], B <= R[rm]
- ALU 操作
  - ALUout <= A + B



多周期数据路径示例



- Critical Path ?

利用分步处理器设计技术

第 1 步：ISA => 逻辑寄存器传输

第 2 步：数据路径的组件

第 3 步：RTL + 组件 => 数据路径

步骤 4：数据路径 + 逻辑 RT => 物理 RT
逻辑 RT 使用 ISA 可见寄存器
物理 RT 也使用中间寄存器

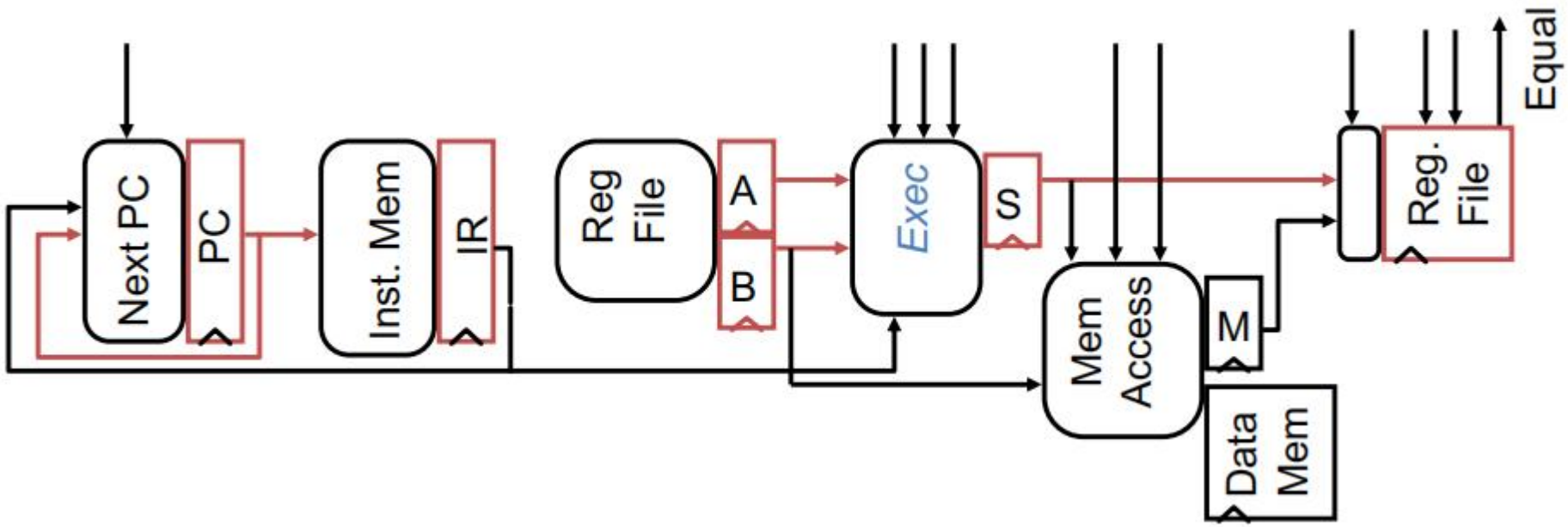
第 5 步：物理 RT => 控制

Step 4: R-rtype (add, sub, . . .)

- Logical Register Transfer
- Physical Register Transfers

inst	Logical Register Transfers
ADD	R[rd] <– R[rn] + R[rm]; PC <– PC + 1

inst	Physical Register Transfers	
	IR <– MEM[pc]	
ADD	A<– R[rn]; B <– R[rm]	
	S <– A + B	
	R[rd] <– S;	PC <– PC + 1



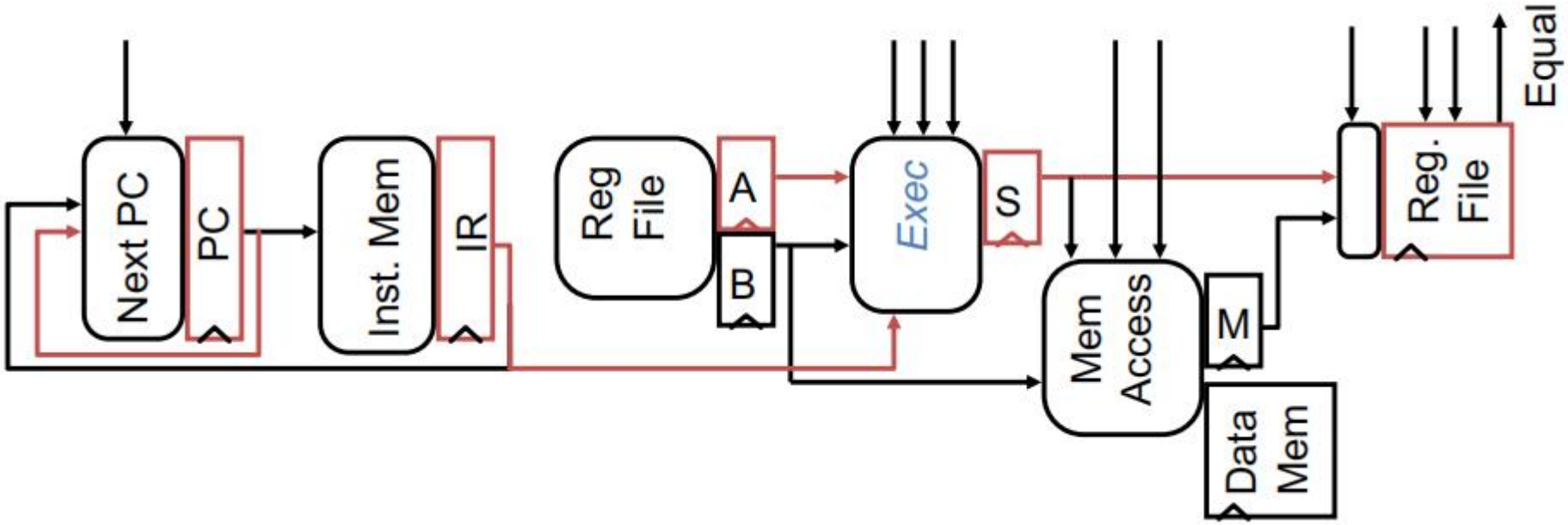
Step 4:Logical immediate

- Logical Register Transfer
- Physical Register Transfers

inst	Logical Register Transfers
ORI	R[rd] <– R[rn] OR ze(imm8); PC <– PC + 1

inst	Physical Register Transfers
	IR <– MEM[pc]
ORI	A<– R[rn]; B <– R[rm]
	S <– A *or* ZeroExt(imm8)
	R[rd] <– S; PC <– PC + 1

side effect

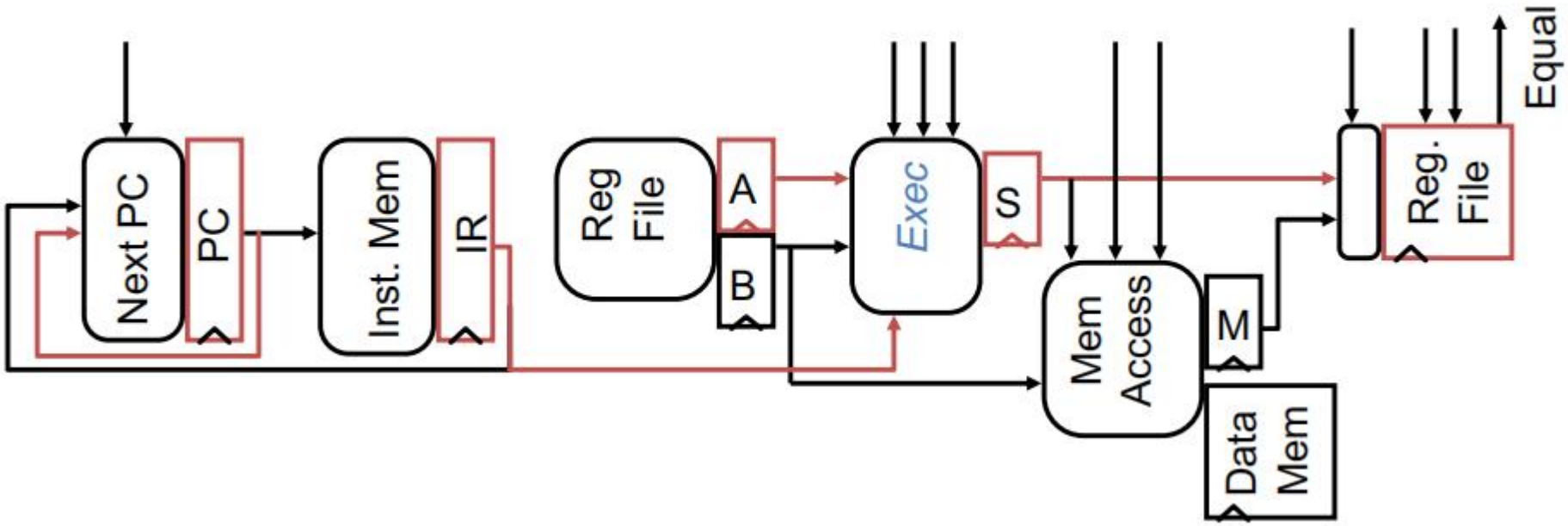


Step 4 : Load

- Logical Register Transfer
- Physical Register Transfers

inst	Logical Register Transfers
LDR	R[rd] <– MEM(R[rn] + sx(imm8); PC <– PC + 1

inst	Physical Register Transfers
	IR <– MEM[pc]
LDR	A<– R[rn]; B <– R[rt]
	S <– A + SignEx(imm8)
	M <– MEM[S]
	R[rd] <– M; PC <– PC + 1



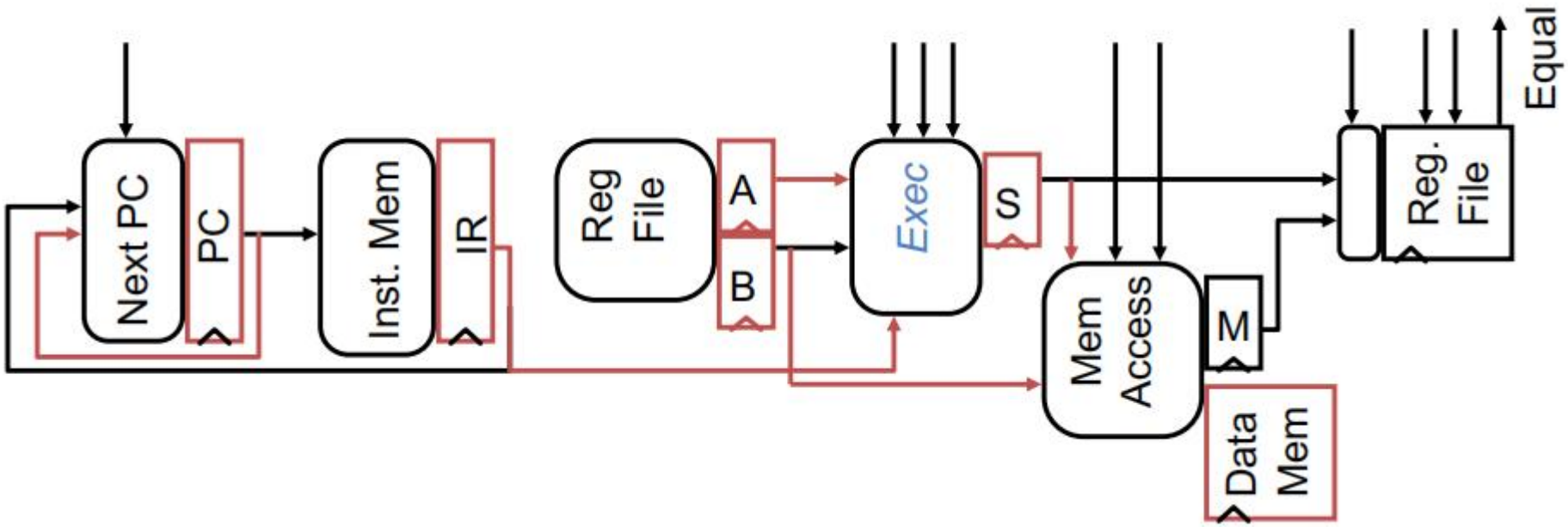
Step 4 : Store

- Logical Register Transfer

inst	Logical Register Transfers
STR	MEM(R[rn] + sx(imm8) <– R[rd]; PC <– PC + 1

- Physical Register Transfers

inst	Physical Register Transfers
	IR <– MEM[pc]
STR	A<– R[rn]; B <– R[rd]
	S <– A + SignEx(imm8);
	MEM[S] <– B PC <– PC + 1



Step 4 : Branch

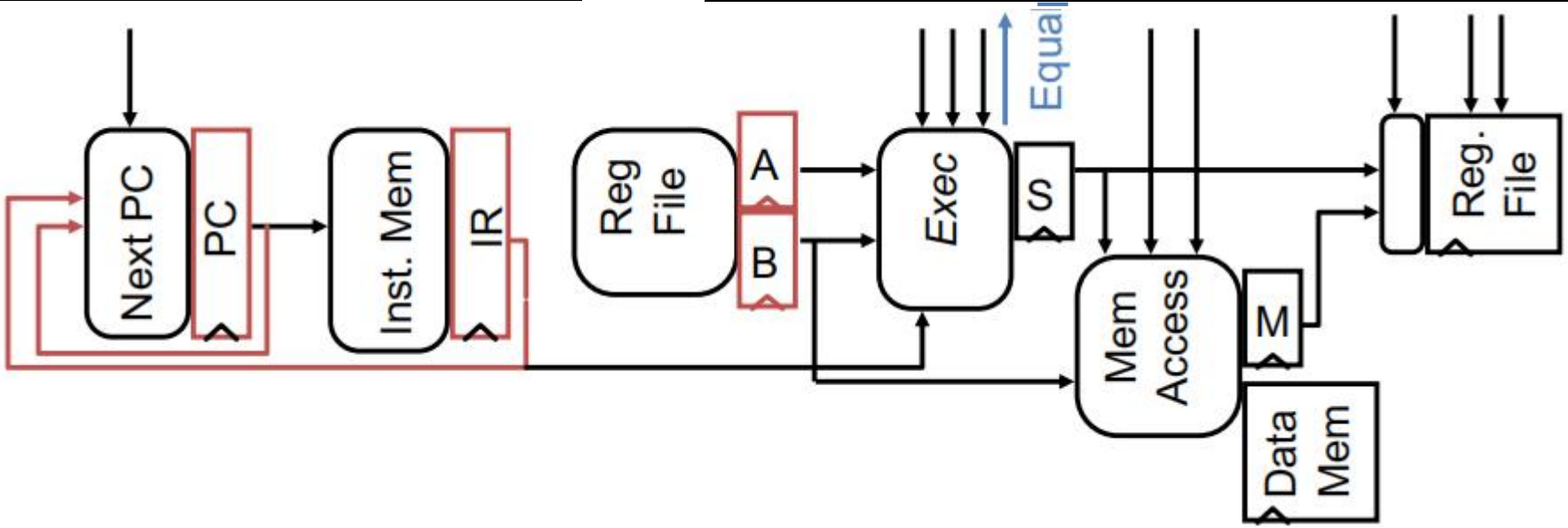
- Logical Register Transfer

inst	Logical Register Transfers
BEQ	if R[rn] == R[rm] then PC <= PC + sx(imm24) else PC <= PC + 1

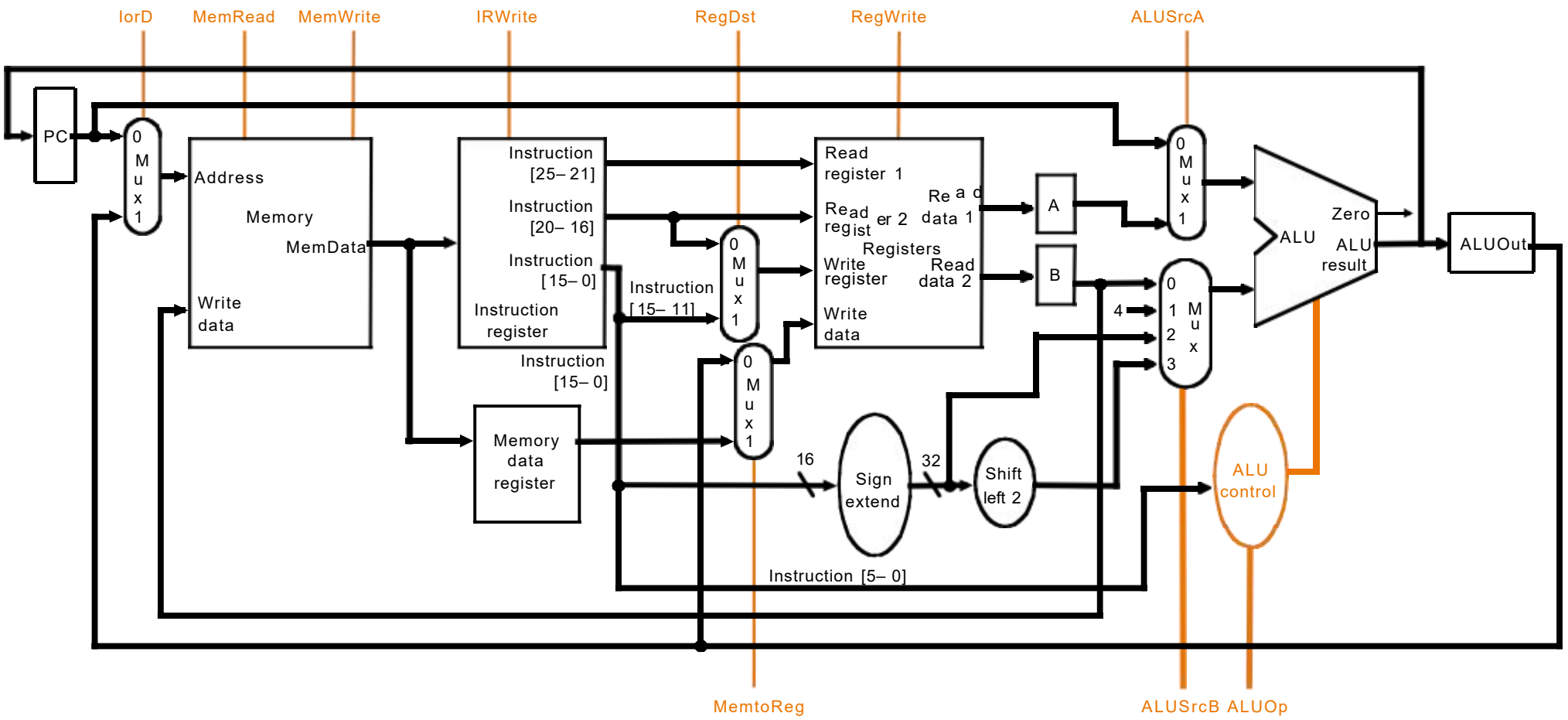
- Physical Register Transfers

inst	Physical Register Transfers	
	IR <– MEM[pc]	
_	A<– R[rn]; B <– R[rm]	
BEQ|Eq	PC <– PC + 1	

inst	Physical Register Transfers	
	IR <– MEM[pc]	
	A<– R[rn]; B <– R[rm]	
BEQ|Eq	PC <– PC + sx(imm16)	



Multiple cycle datapath with control lines



Step 4
Control Specification for multicycle datapath