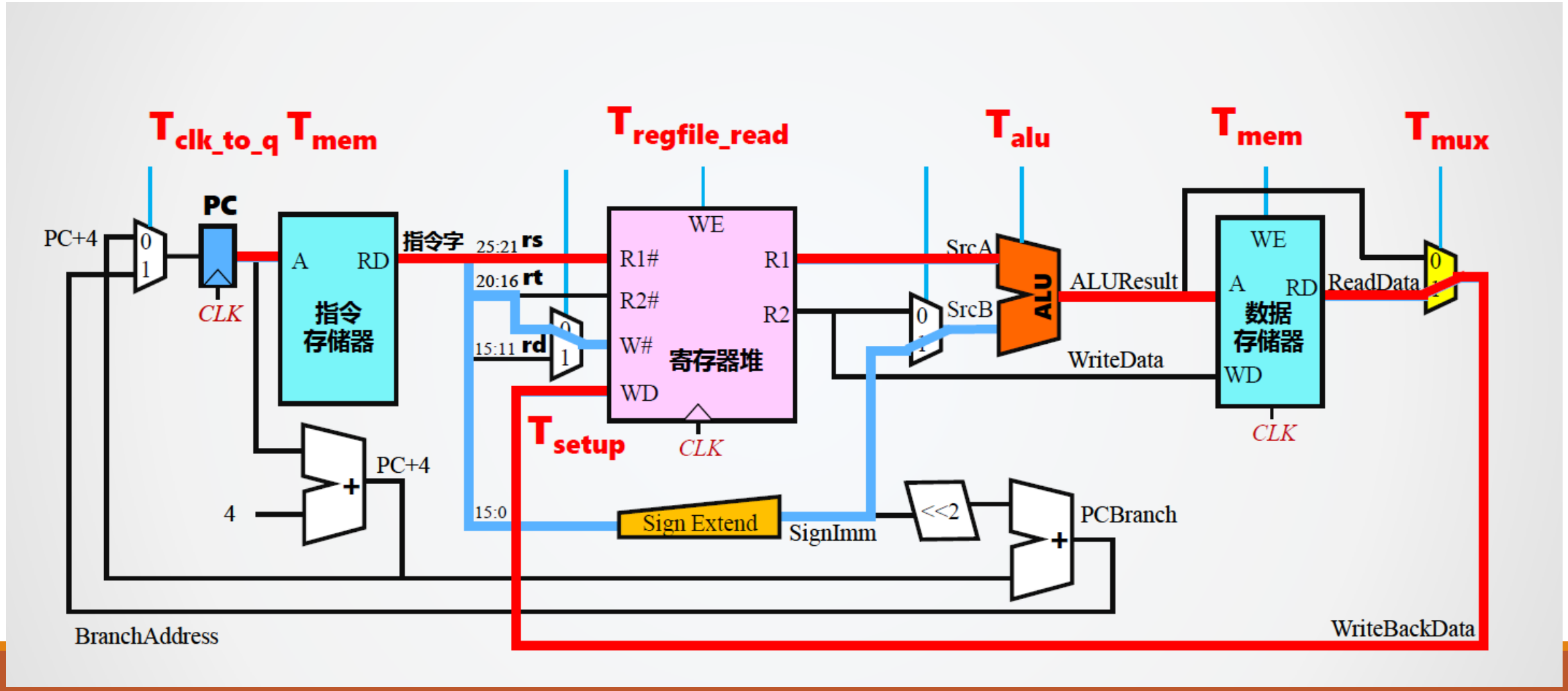
单周期MIPS CPU存在的问题

单个时钟周期时间过长
CPU整体性能取决于最慢的指令

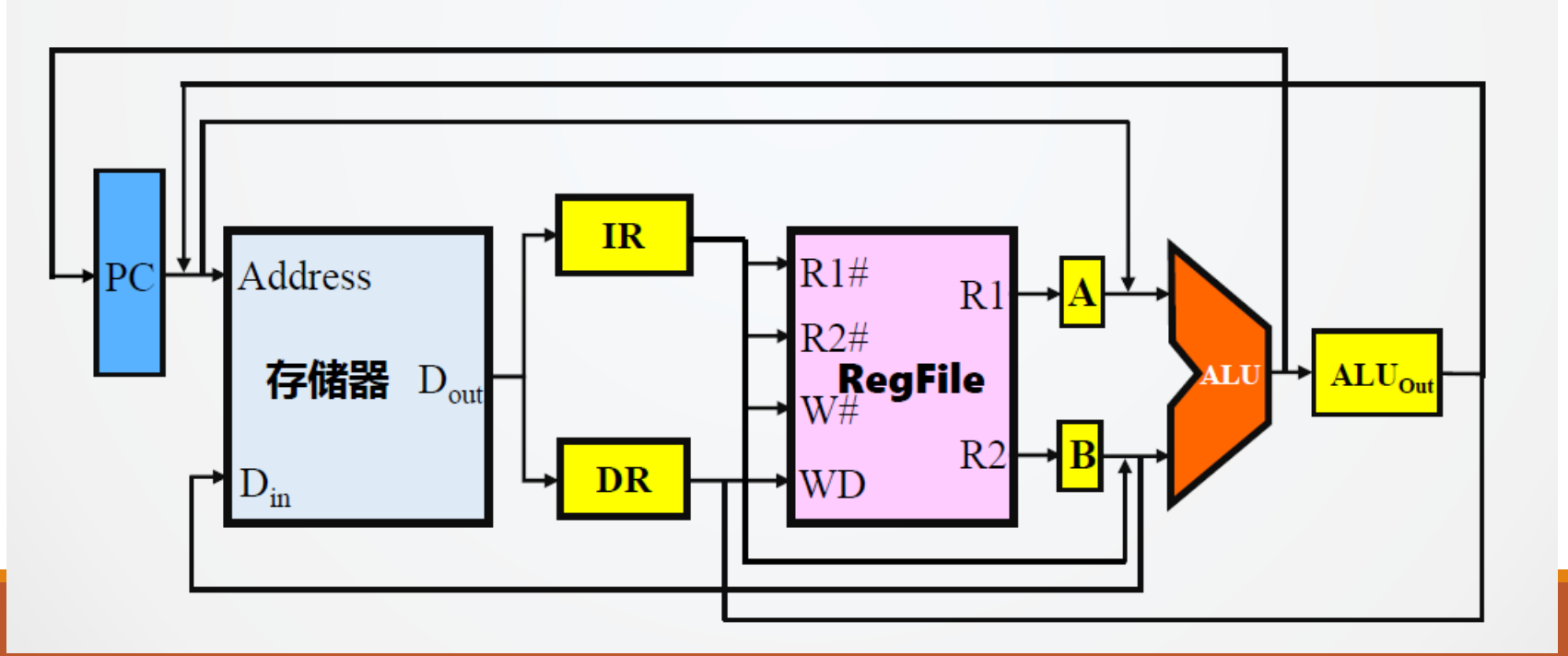
LW指令的关键路径时长分析



如何解决？——多周期的MIPS CPU

多周期MIPS CPU的特点

- 不再区分指令和数据存储器，设置统一存储器，分时复用
- 功能部件的输出端增加寄存器，用于锁存数据
- 分多个周期，时钟周期相对变小，传输通路变短



选做任务1

二、多周期的MIPS CPU设计实验（8条指令）

实验目标

- 熟悉多周期MIPS CPU设计原理
- 能利用相关原理在Logisim中设计实现MIPS多周期CPU，支持内存中简单的冒泡排序程序

主要任务（设计思路）

- 设计绘制多周期MIPS CPU的数据通路
- 实现多周期硬布线控制器
- 连接控制器与数据通路
- 软、硬件联调测试

与单周期MIPS CPU的区别

能支持下列8条指令组成的程序

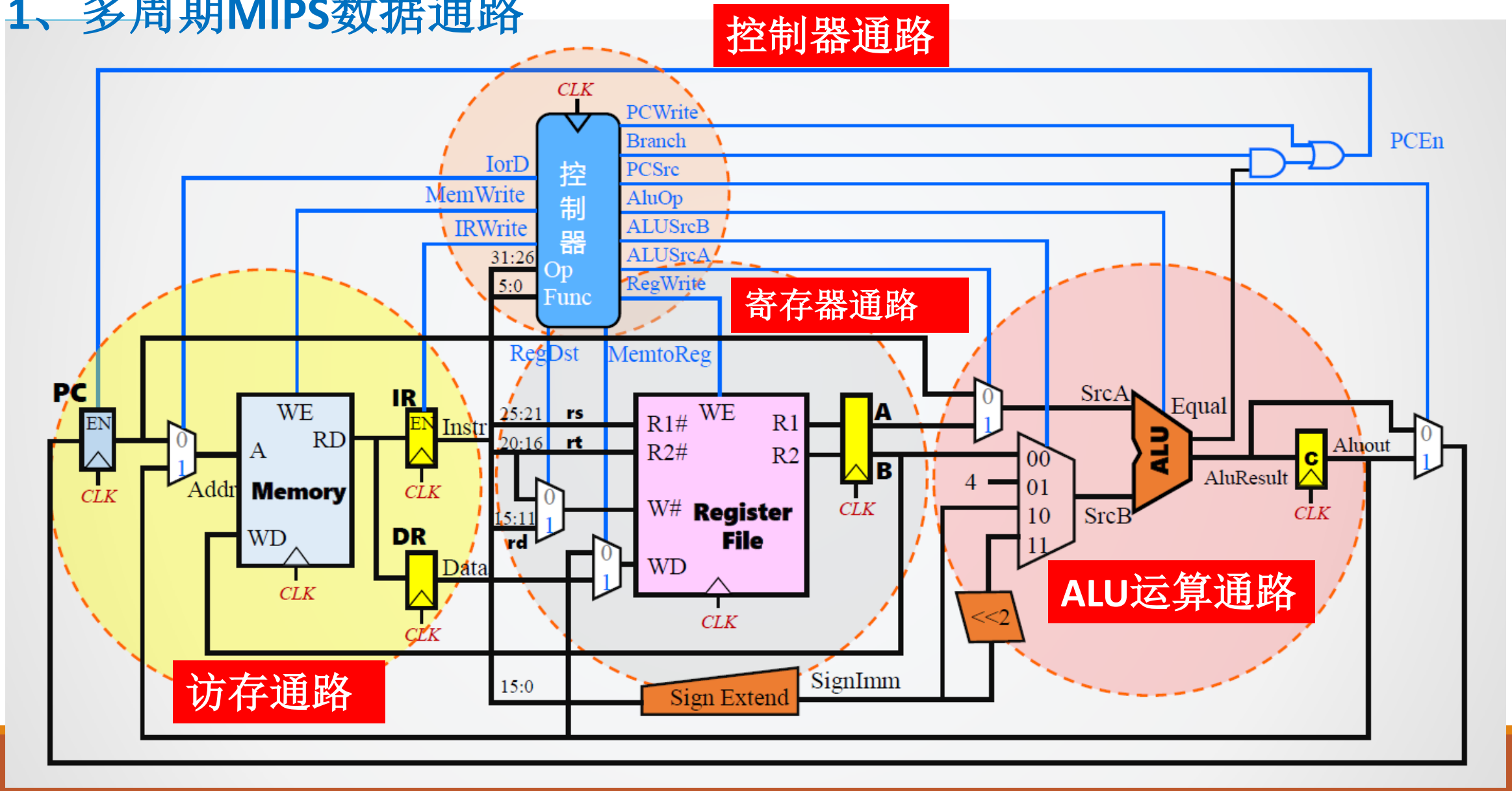
本次实验设计的原型系统，不处理溢出处理

#	MIPS指令	RTL功能描述
1	add $rd,$rs,$rt	R[$rd]←R[$rs]+R[$rt]　溢出时产生异常，且不修改R[$rd]
2	slt $rd,$rs,$rt	R[$rd]←R[$rs]<R[$rt]　小于置1，有符号比较
3	addi $rt,$rs,imm	R[$rt]←R[$rs]+$SignExt_{16b}$(imm)　溢出产生异常
4	lw $rt,imm($rs)	R[$rt]←$Mem_{4B}$(R[$rs]+$SignExt_{16b}$(imm))
5	sw $rt,imm($rs)	Mem_{4B}(R[$rs]+$SignExt_{16b}$(imm))←R[$rt]
6	beq $rs,$rt,imm	if(R[$rs] = R[$rt]) PC ← PC + $SignExt_{18b}$({imm, 00})
7	bne $rs,$rt,imm	if(R[$rs] != R[$rt]) PC ← PC + $SignExt_{18b}$({imm, 00})
8	syscall	系统调用，这里用于停机

本次实验只用到其特殊的停机功能

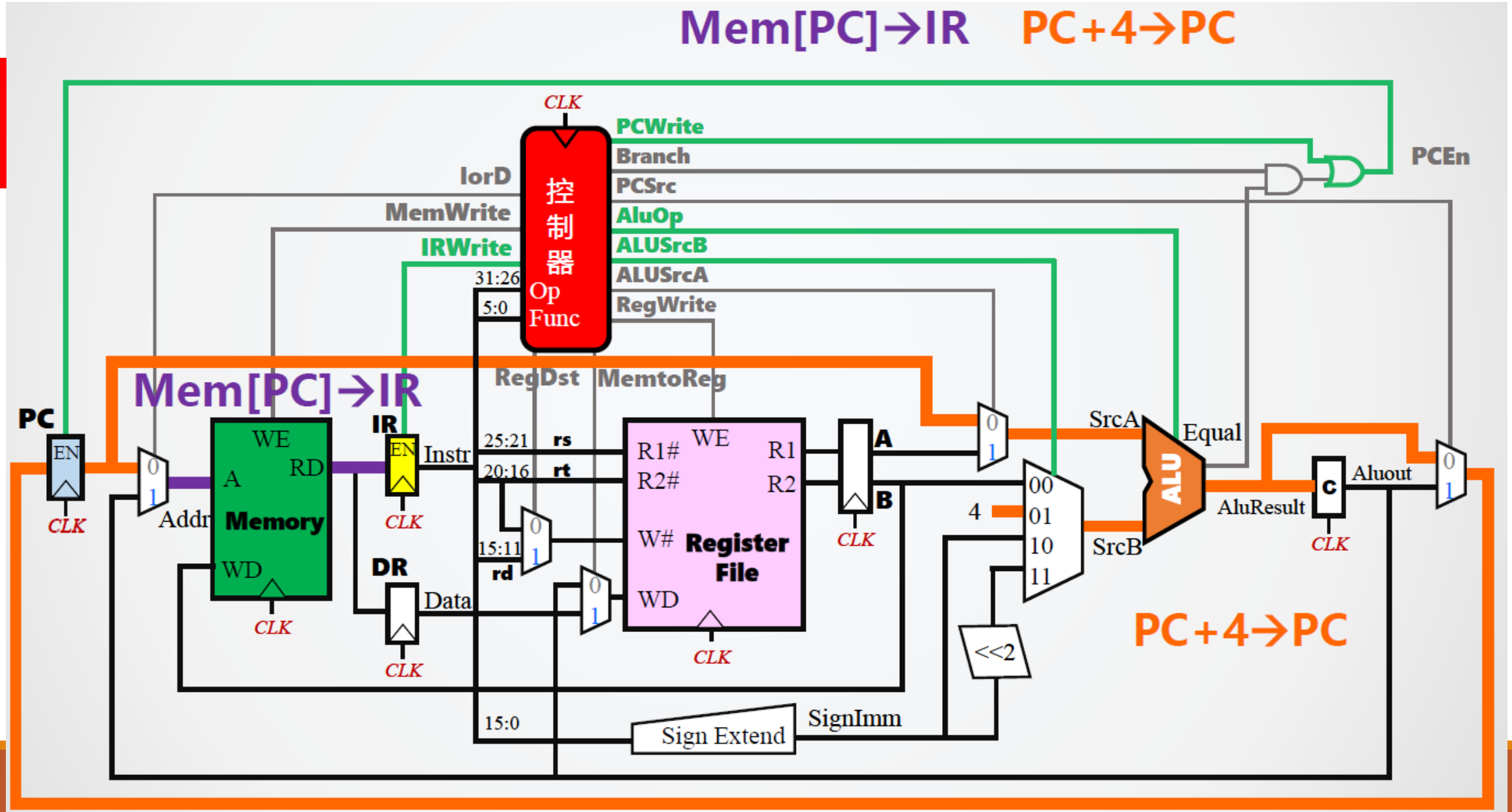
多周期MIPS CPU的设计原理

1、多周期MIPS数据通路



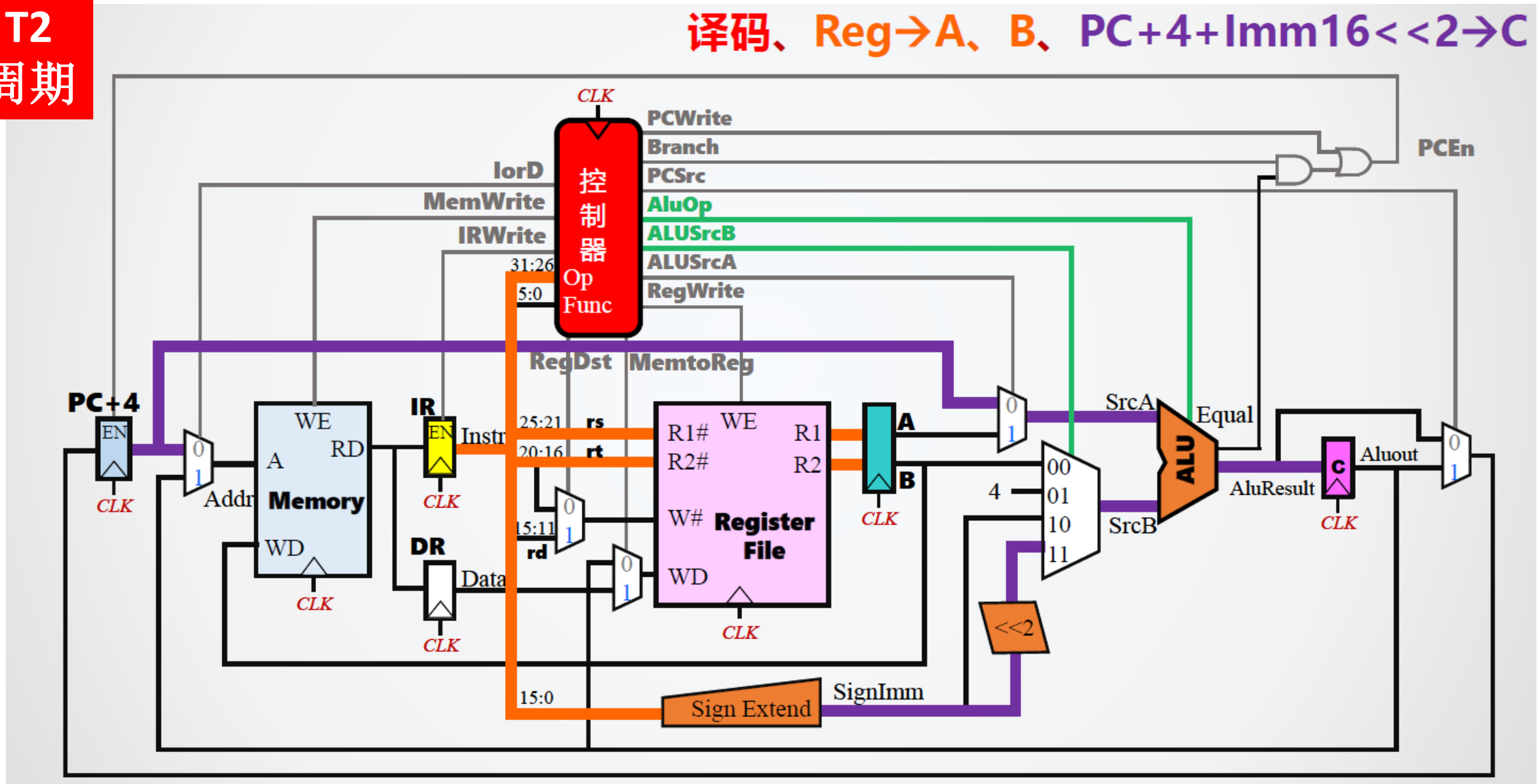
2、多周期MIPS CPU中典型指令不同阶段的数据通路建立

T1 周期



T2
周期

译码、Reg→A、B、PC+4+Imm16<<2→C

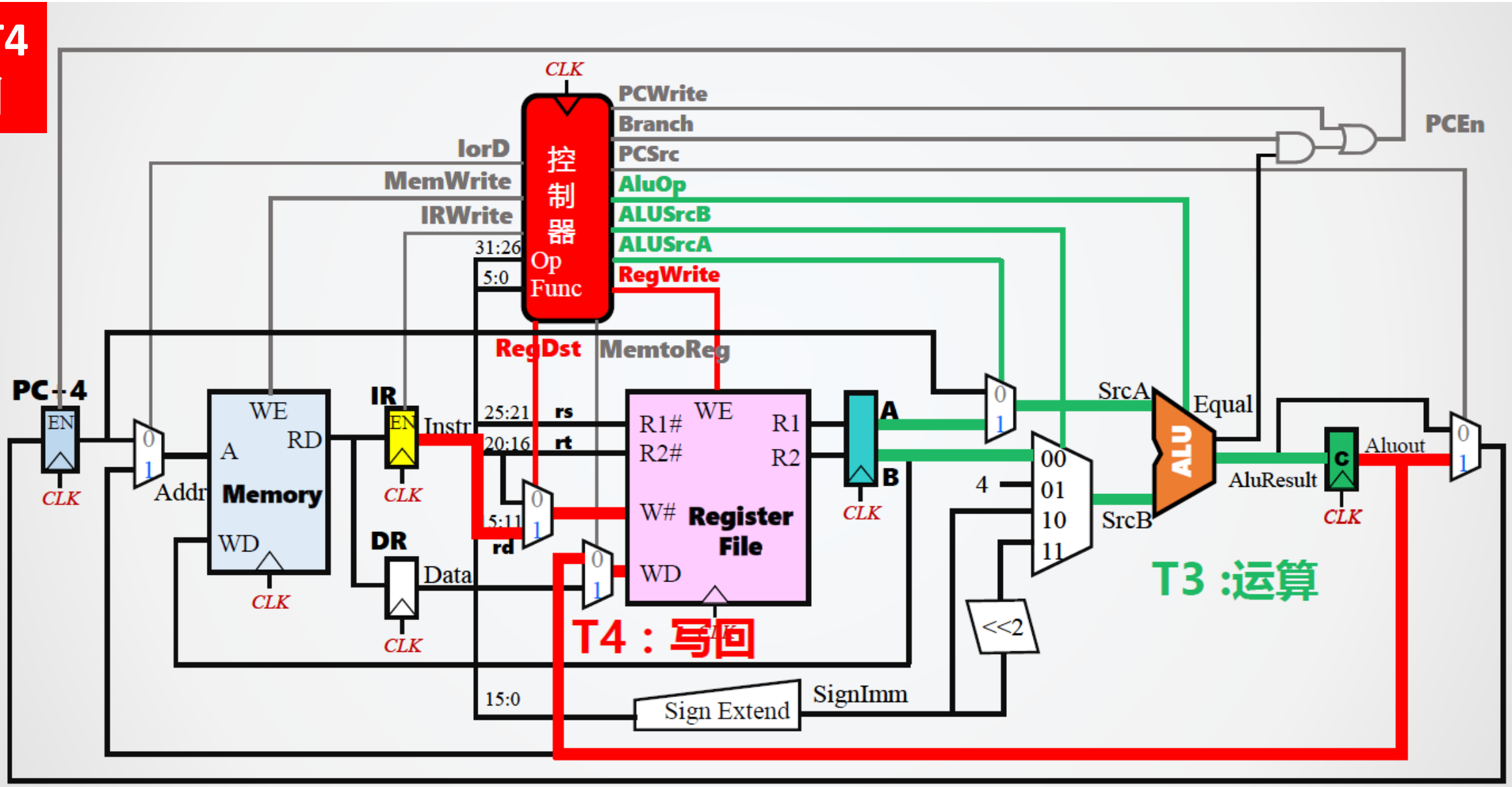


T1、T2是所有指令的共有阶段，可视为取指令阶段

◆ R型指令

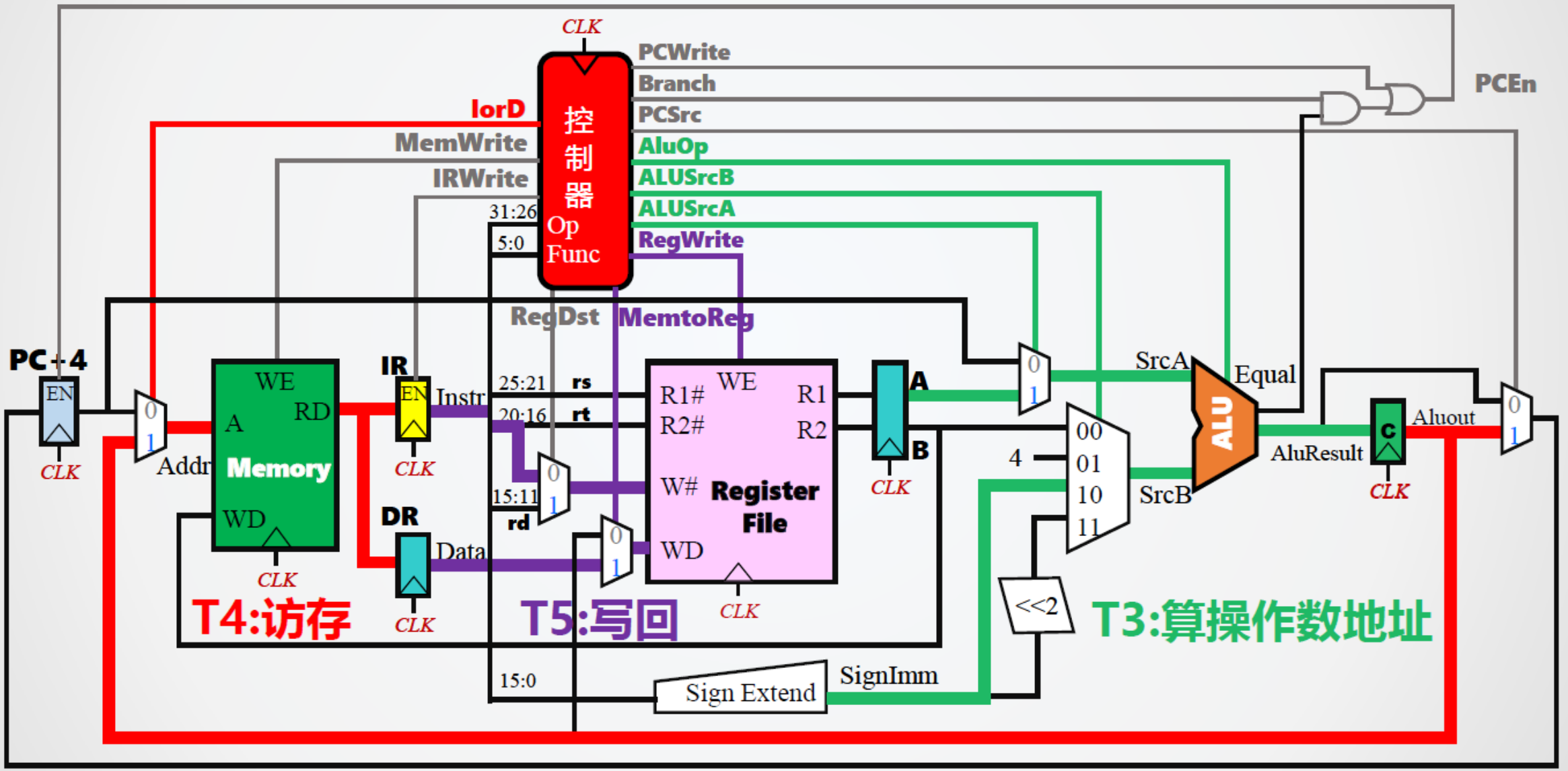
不同指令的执行阶段所需周期不同

T3、T4 周期



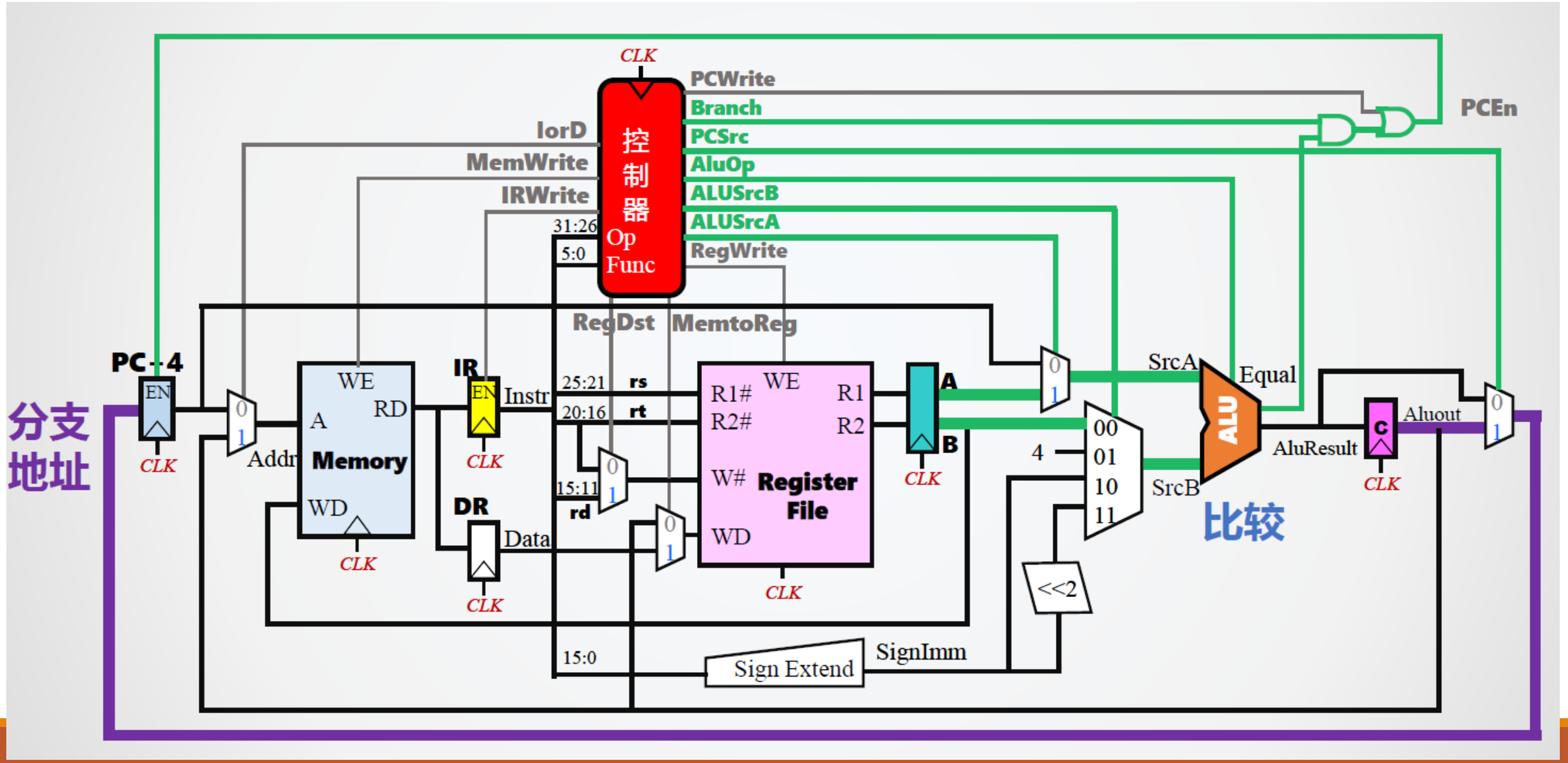
◆ LW指令

T3、T4 、T5周期



◆ BEQ指令

T3周期


CLK
PCWrite
Branch
PCSrc
AluOp
ALUSrcB
ALUSrcA
RegWrite
RegDst
MemtoReg
IorD
MemWrite
IRWrite
31:26
5:0
Op
Func
控制器
PCEn
PC+4
EN
CLK
Addr
WE
RD
A
Memory
WD
IR
Instr
DR
Data
25:21
rs
20:16
rt
15:11
rd
R1#
R2#
W#
WD
R1
R2
Register File
A
B
SrcA
SrcB
ALU
Equal
Aluout
AluResult
C
4
00
01
10
11
<<2
15:0
Sign Extend
SignImm
分支地址
比较

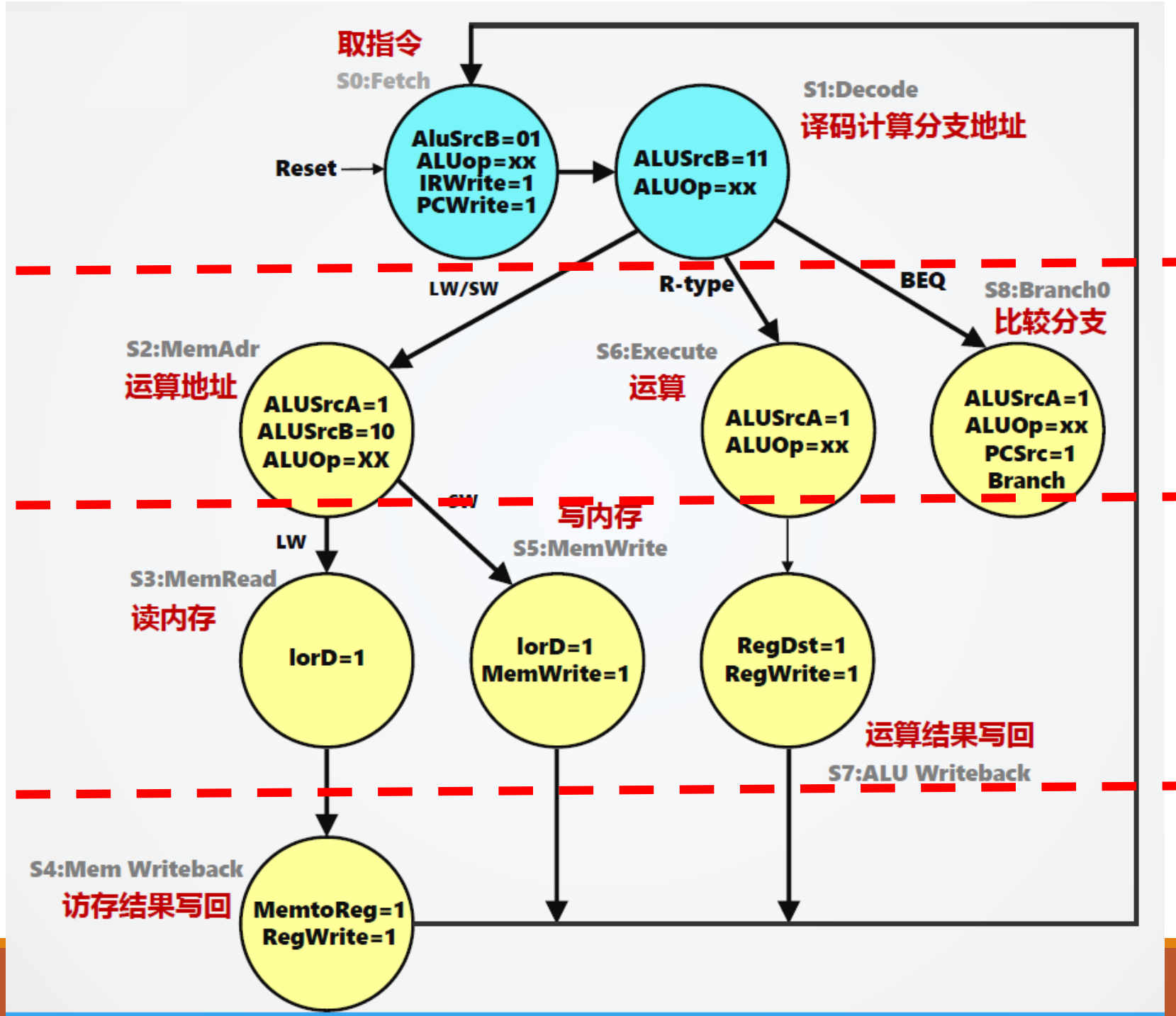
3、多周期MIPS CPU的指令执行流程总结

T1、T2

T3

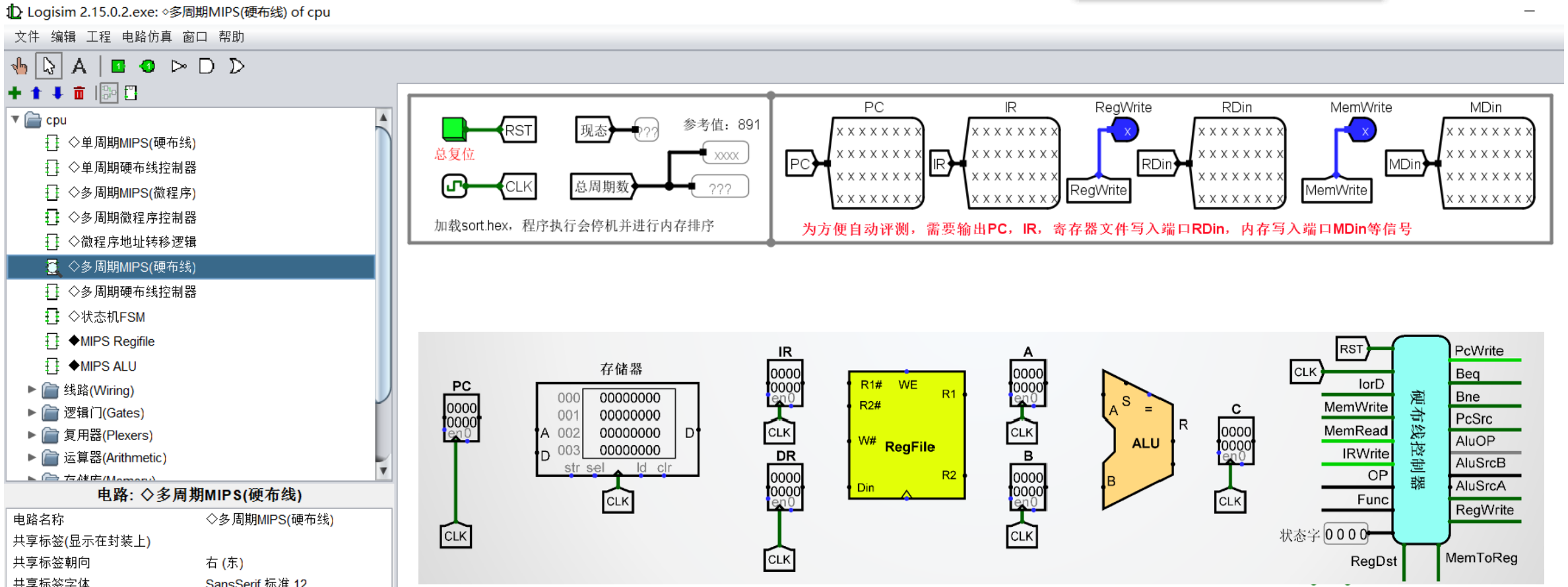
T4

T5

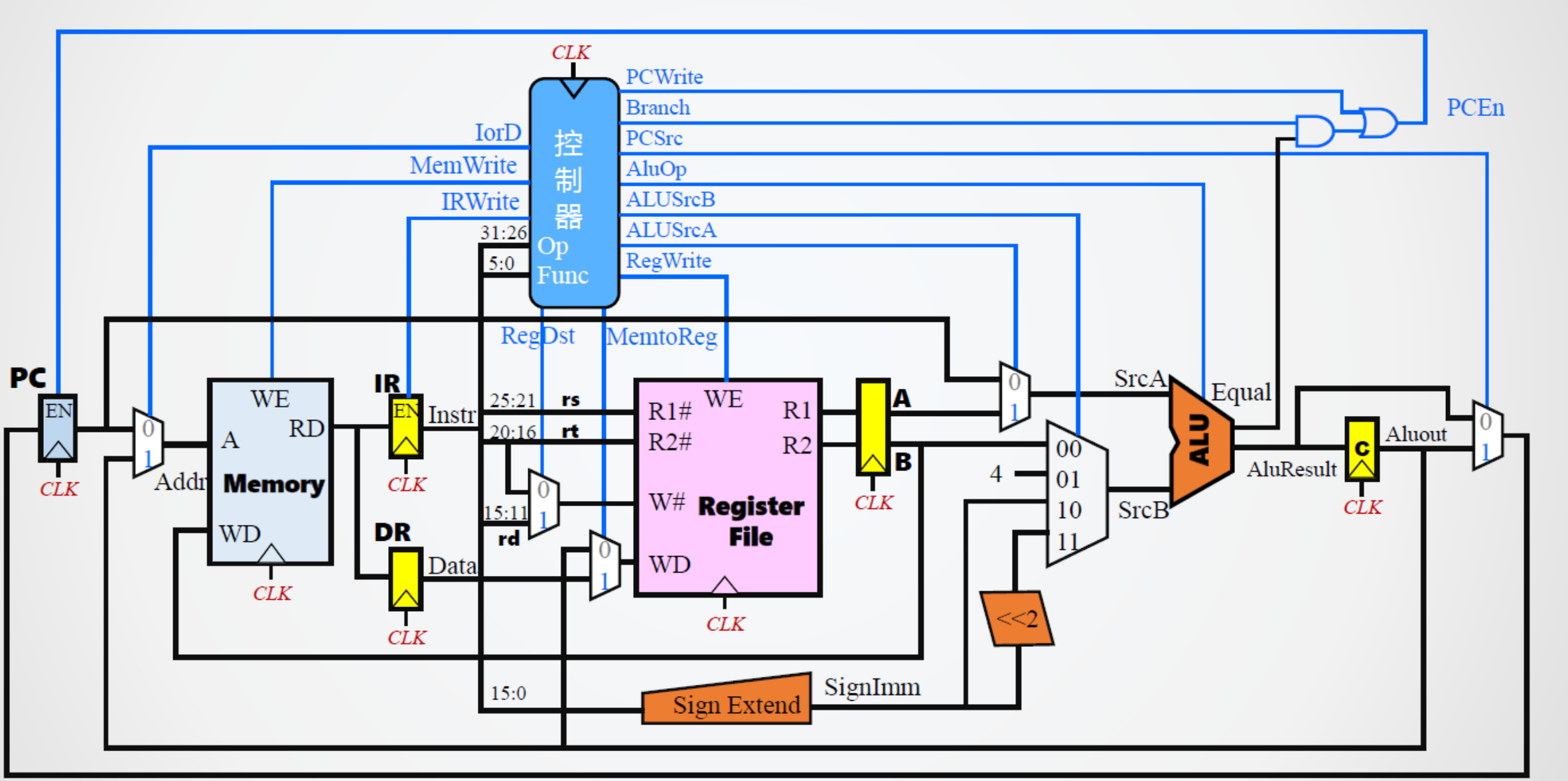


步骤（一）：多周期MIPS的数据通路设计

打开cpu.circ文件中，选中“多周期MIPS（硬布线）”，利用已经完成的标准模块，并添加多路选择器、寄存器等其他组件构成数据通路

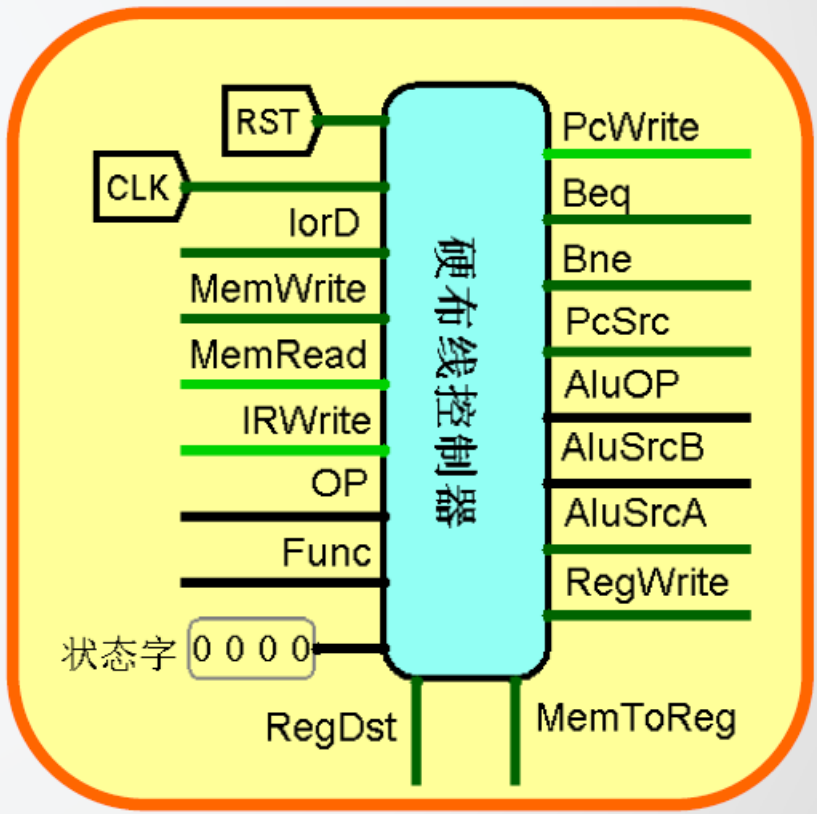


参考的数据通路

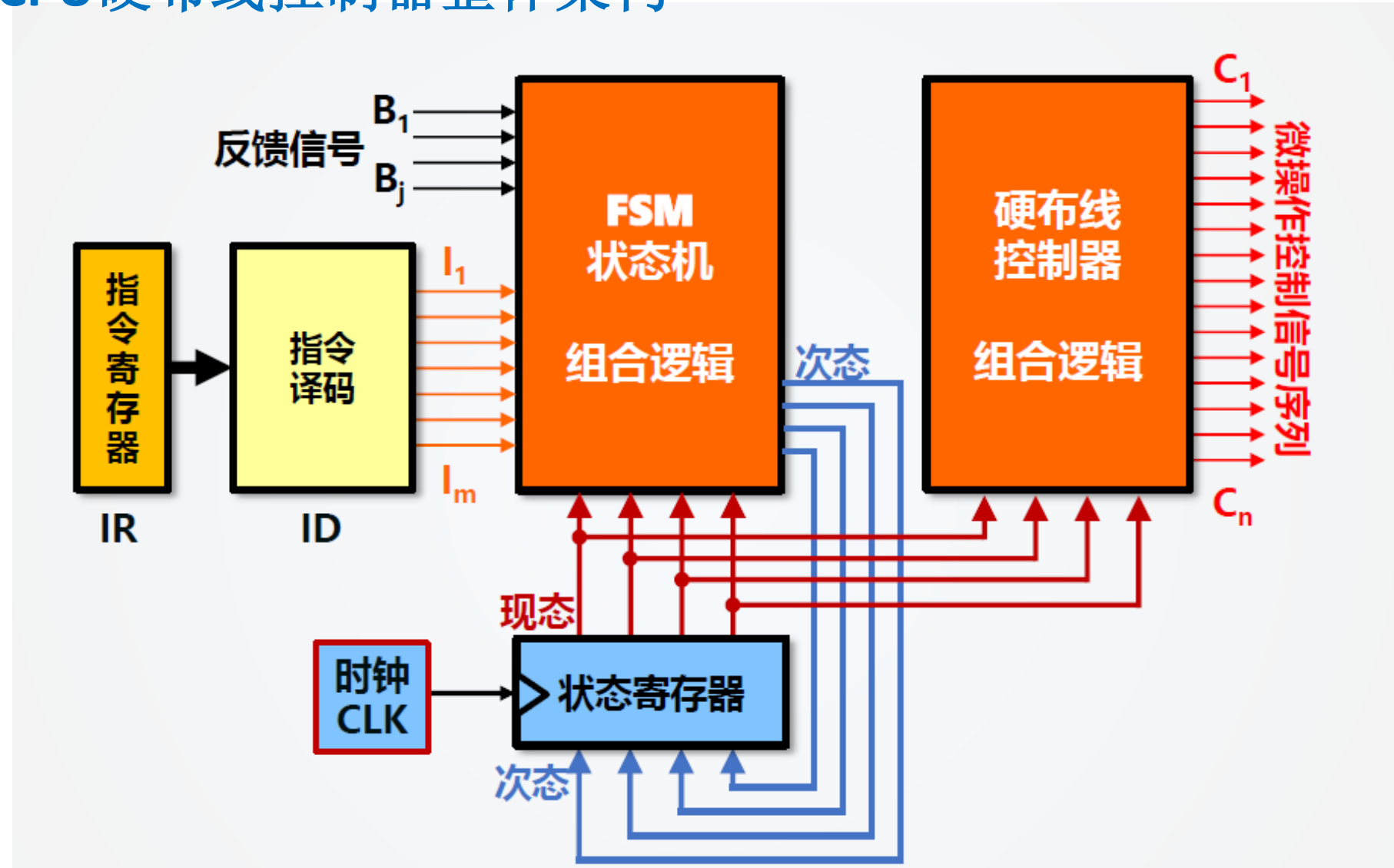

CLK
PCWrite
Branch
PCSrc
AluOp
ALUSrcB
ALUSrcA
RegWrite
PCEn
IorD
MemWrite
IRWrite
31:26
5:0
Op
Func
控制器
RegDst
MemtoReg
PC
EN
CLK
Addr
WE
RD
A
Memory
WD
IR
Instr
DR
Data
25:21
rs
20:16
rt
15:11
rd
R1#
R2#
W#
WD
R1
R2
Register File
A
B
SrcA
SrcB
ALU
Equal
AluResult
C
Aluout
4
00
01
10
11
<<2
15:0
Sign Extend
SignImm

步骤（二）：多周期MIPS控制器的设计

- 输入信号
  - 指令字Opcode，Func字段（12位）
  - 时钟信号、复位信号
- 输出信号
  - 多路选择器选择信号
  - 内存访问控制信号
  - 寄存器写使能信号
  - 运算器控制信号、指令译码信号

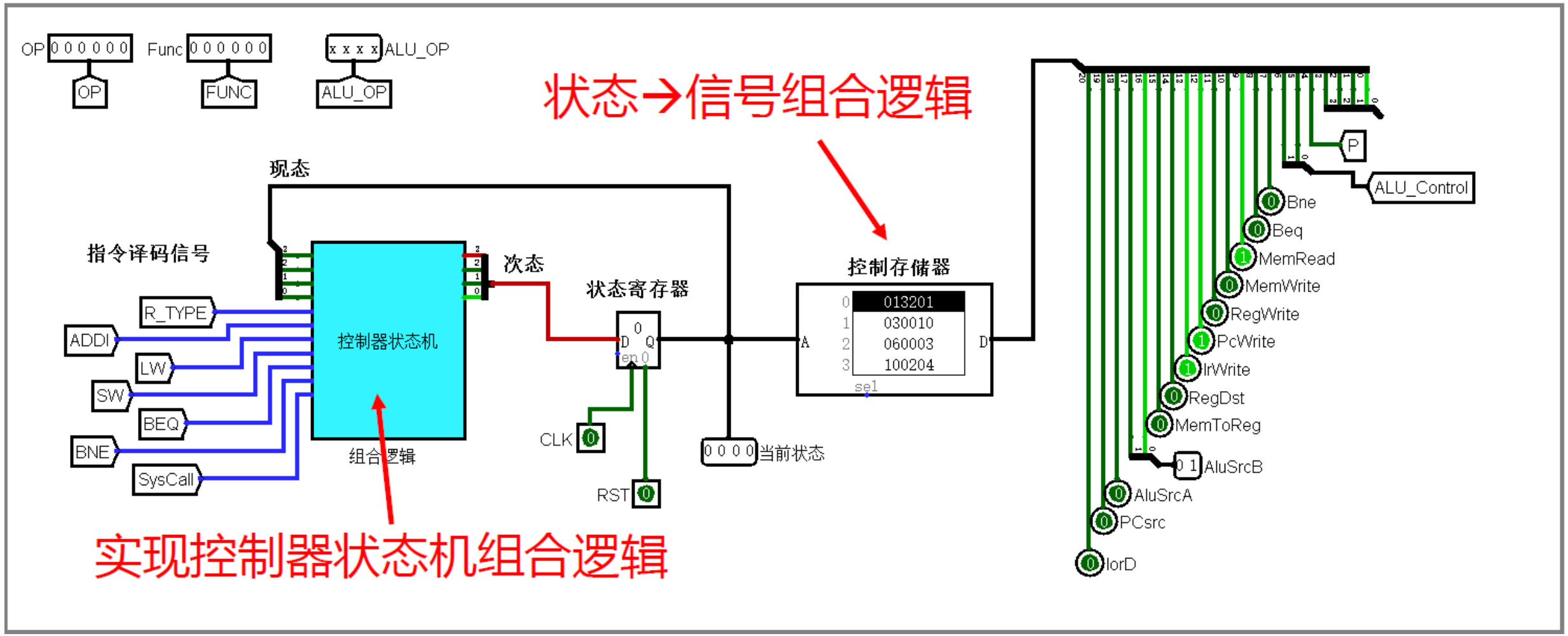


多周期MIPS CPU硬布线控制器整体架构



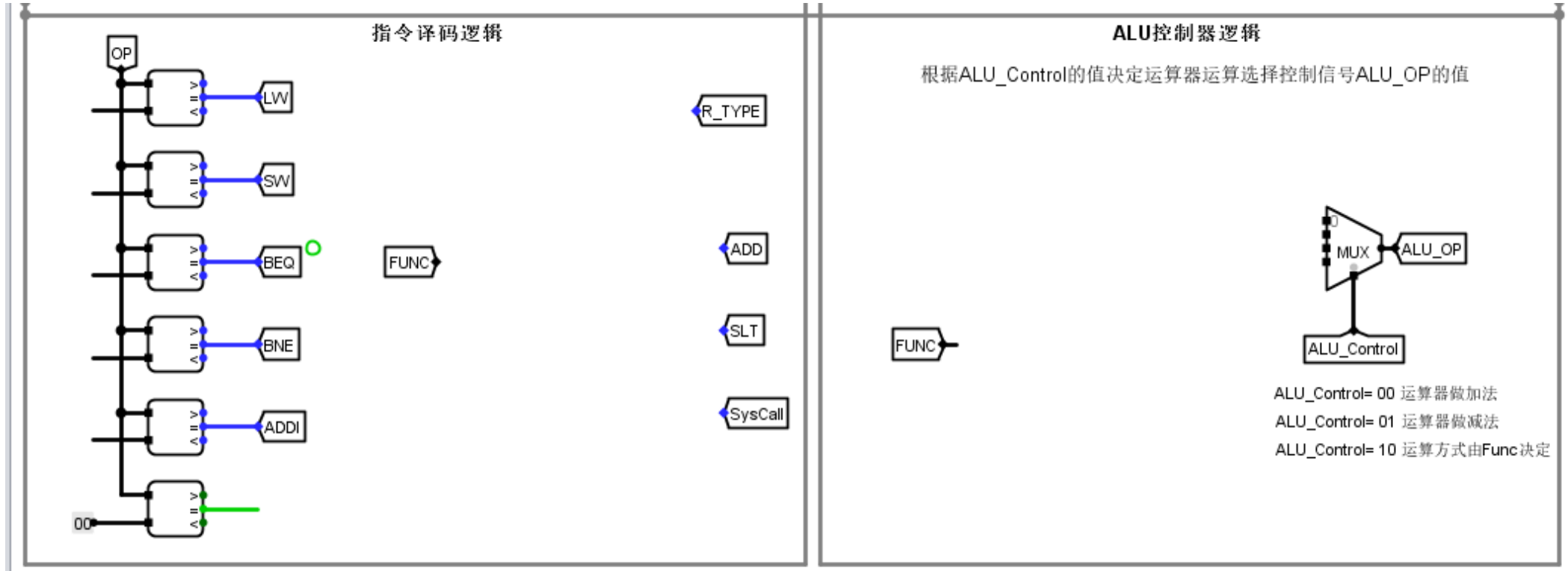
机器指令字 → 控制器信号序列

硬布线控制器的内部架构



1、完善硬布线控制器组合逻辑的内部电路

- 打开“多周期硬布线控制器”
- 实现指令译码、ALU控制逻辑
- ALU_Control 的设置是为了简化控制器的设计



给出简单的逻辑实现对应指令译码信号，LW、SW、BEQ、BNE、ADDI、ADD、SLT、SYSCALL、R_TYPE。

注意R_TYPE表示R型运算指令，SYSCALL是特殊的R型指令，不属于这个类别

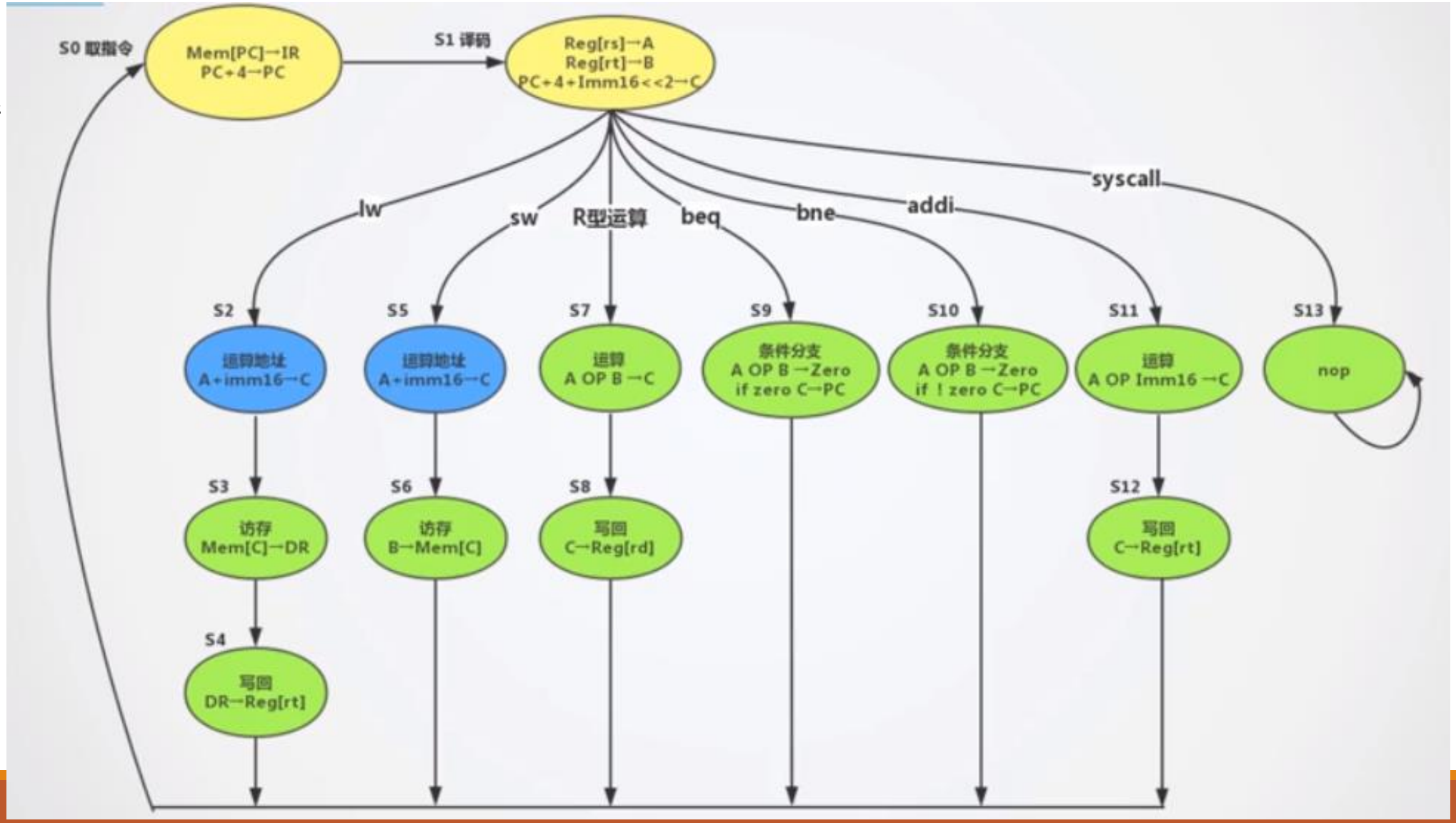
控制信号产生条件分析

#	控制信号	信号说明	产生条件
1	PCWrite	PC写使能控制	取指令周期，分支指令执行
2	IorD	指令还是数据	0表示指令，1表示数据
3	IRwrite	指令寄存器写使能	高电平有效
4	MemWrite	写内存控制信号	sw指令
5	MemRead	读内存控制信号	lw指令 取指令
6	Beq	Beq指令译码信号	Beq指令
7	Bne	Bne指令译码信号	Bne指令
8	PcSrc	PC输入来源	顺序寻址还是跳跃寻址
9	AluOP	运算器操作控制符 4位	ALU_Control控制，00加，01减，10由Funct定
10	AluSrcA	运算器第一输入选择	
11	AluSrcB	运算器第二输入选择	Lw指令，sw指令，addi
12	RegWrite	寄存器写使能控制信号	寄存器写回信号
13	RegDst	写入寄存器选择控制信号	R型指令
14	MemToReg	写入寄存器的数据来自存储器	lw指令

2、控制器状态机组合逻辑电路的设计实现

（1）构建控制器状态转换图

◆ 共14个状态



（2）借助Excel表格，完成状态机逻辑的自动生成

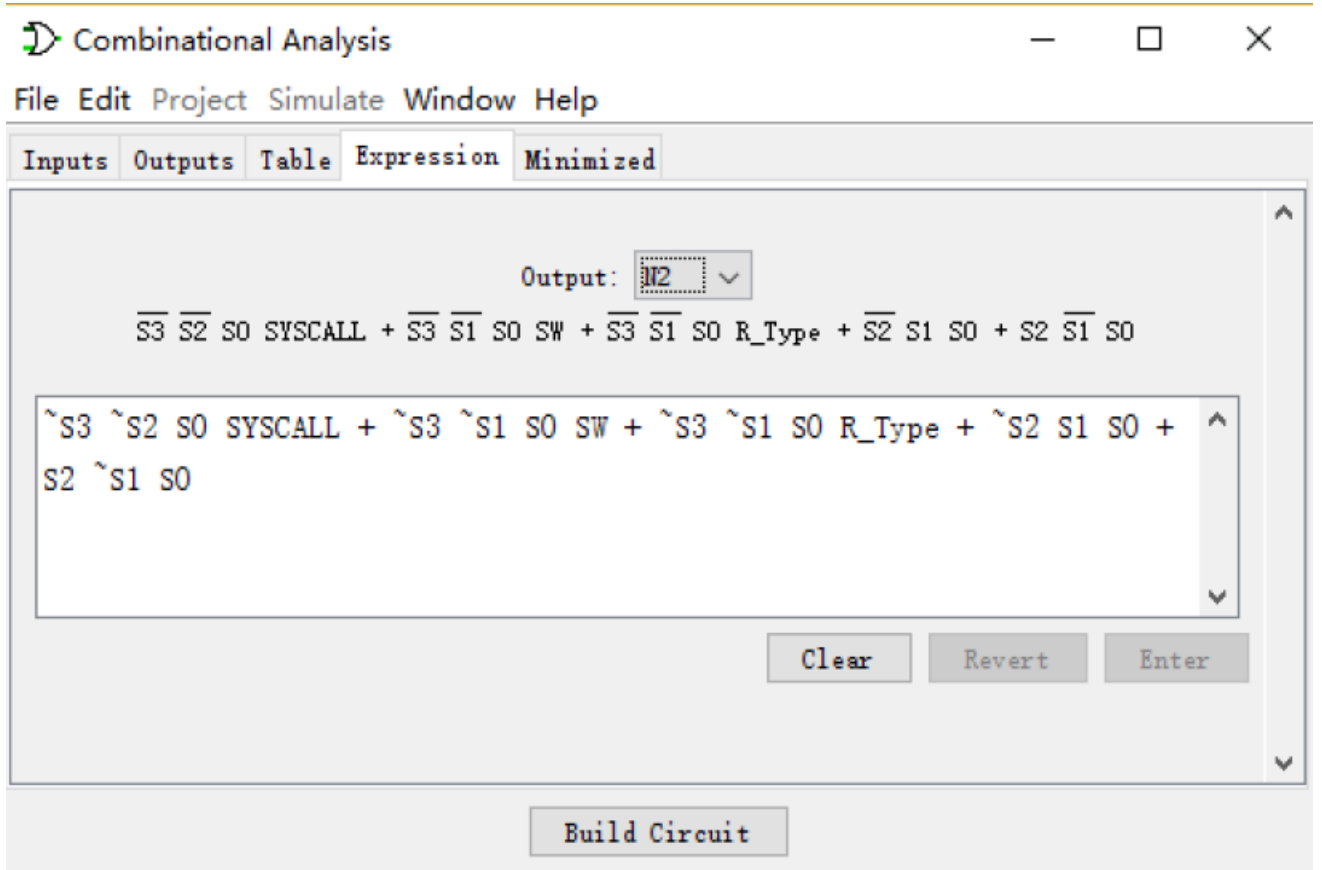
- **填写状态转换表**　（输入：现态，指令译码信号，输出次态）
- **自动生成次态逻辑表达式**　（复制表达式到Logisim）

S3	S2	S1	S0	最小项表达式	N3	N2	N1	N0
~S3&	~S2&	~S1&	~S0&	~S3&~S2&~S1&~S0&R_Type&SW	~S3&~S2&~S1&~S0&R_Type&SW+			~S3&~S2&~S1&~S0&R_Type&SV+
~S3&	~S2&	~S1&	S0&	~S3&~S2&~S1&S0&R_Type		~S3&~S2&~S1&S0&R_Type+	~S3&~S2&~S1&S0&R_Type+	~S3&~S2&~S1&S0&R_Type+
~S3&	~S2&	~S1&	S0&	~S3&~S2&~S1&S0&LW			~S3&~S2&~S1&S0&LW+	
				逻辑表达式->>>	~S3&~S2&~S1&~S0&R_Type&SW	~S3&~S2&~S1&S0&R_Type		

状态转换表　表达式自动生成

（3）生成状态机组合逻辑电路

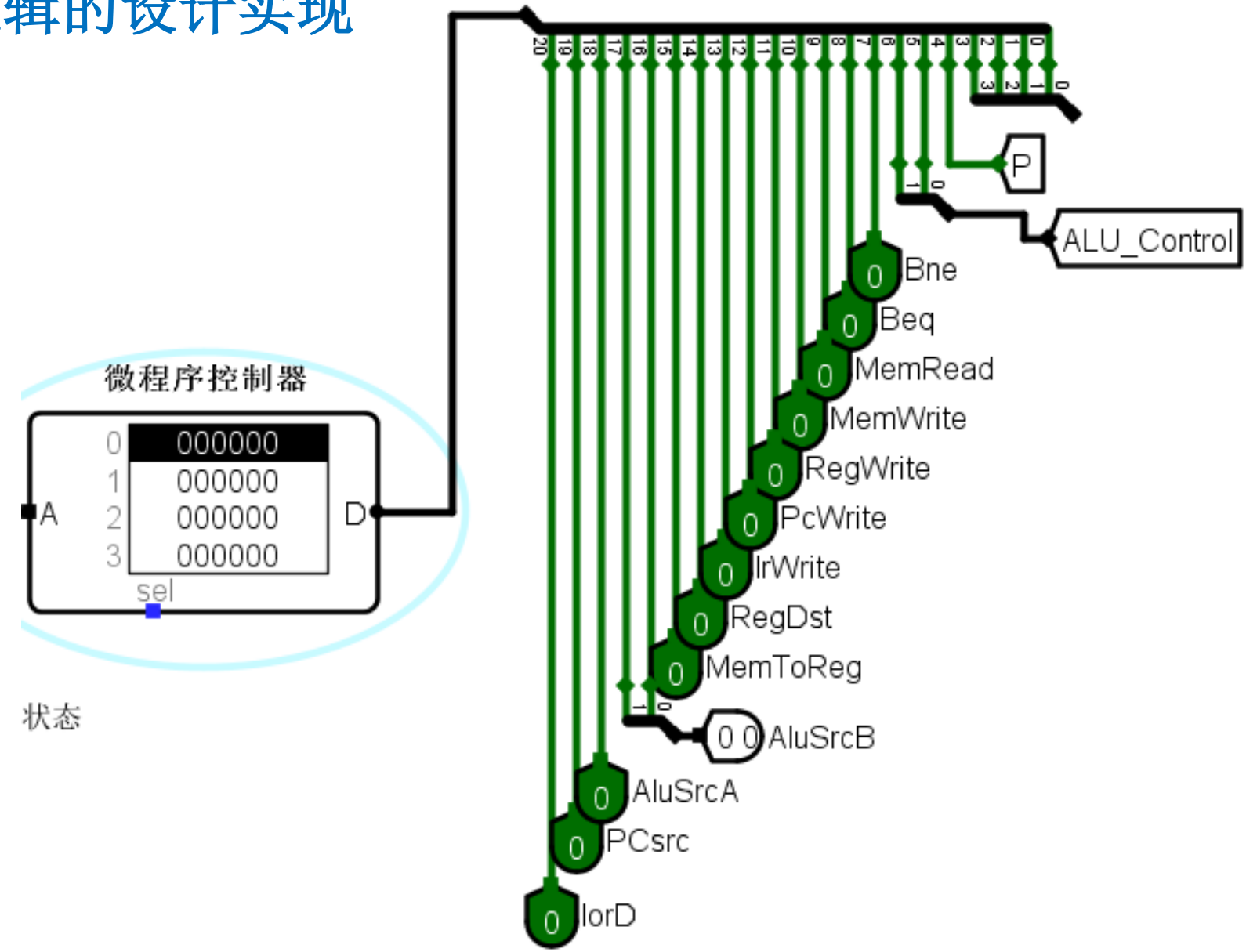
- 打开状态机FSM子电路
- Project→Analyze Circuit
- 输入对应状态机各位表达式
- 自动生成电路



3、控制器状态信号生成逻辑的设计实现

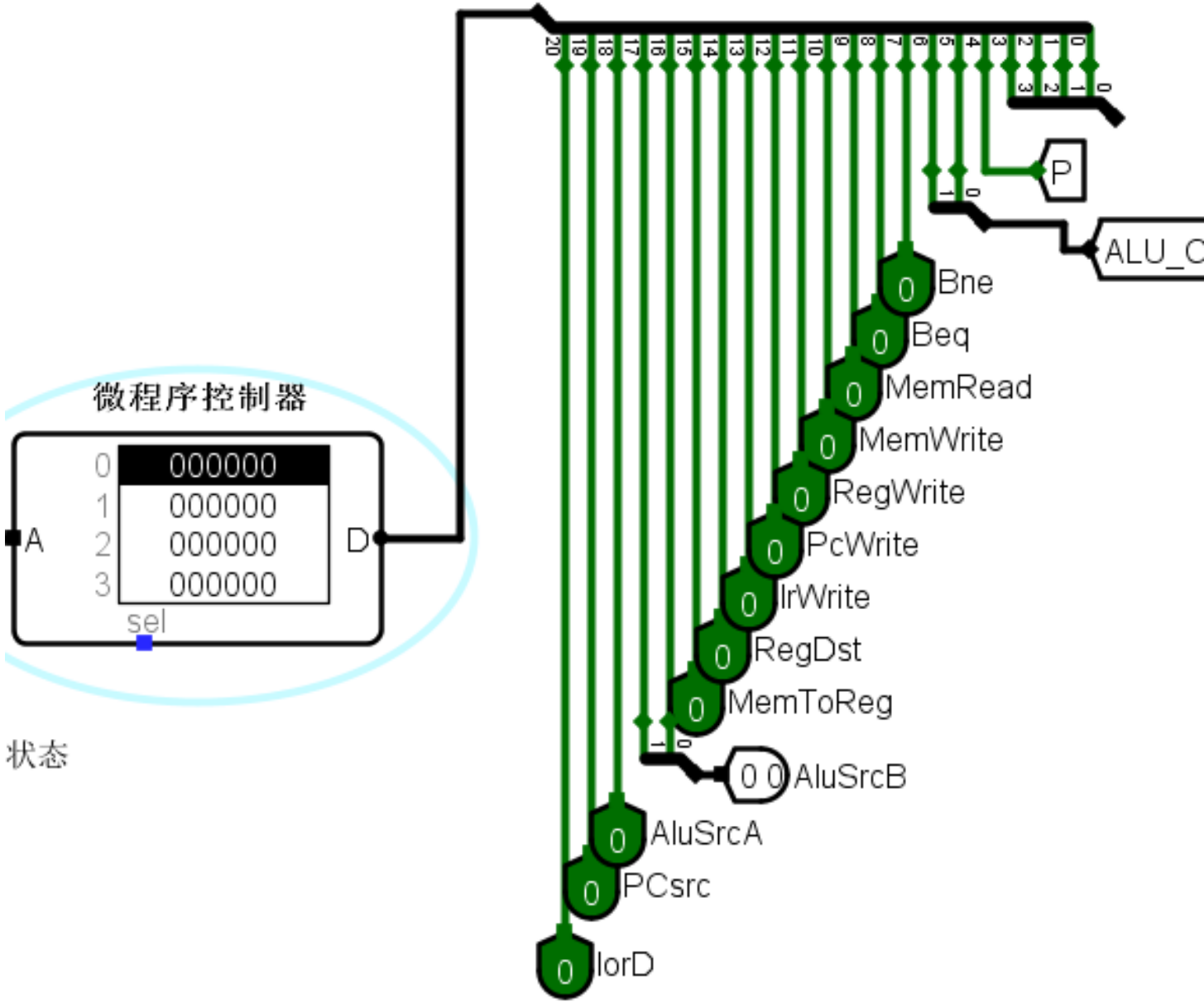
借助微程序控制器的思想，为每个状态对应要生成的控制信号序列进行编码

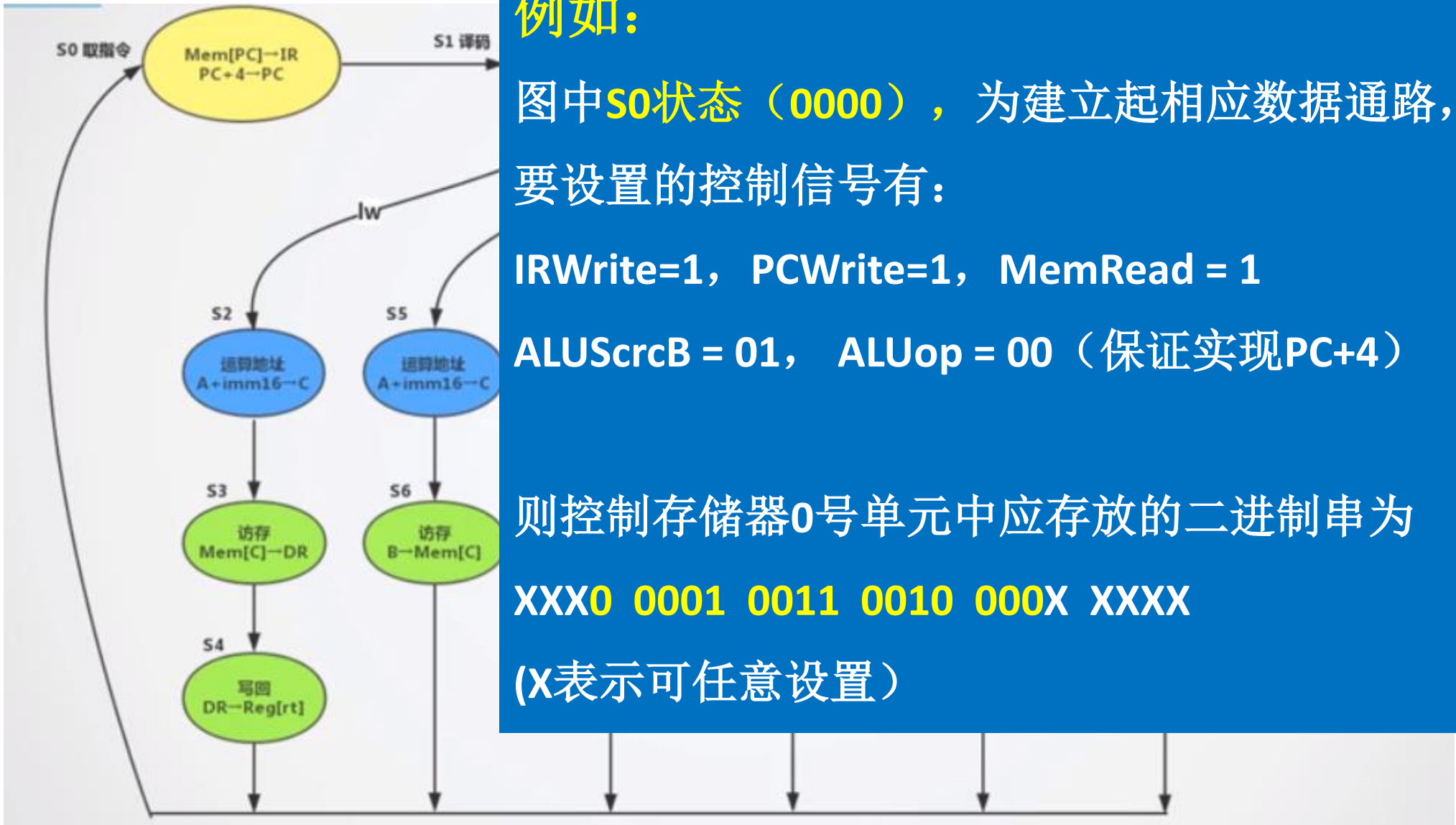
状态编码即对应微程序控制器的地址码



微程序控制器每个单元存放24bit，其中：

- 23-21位并未使用，可任意设置；
- 4-0位在微程序控制器设计中使用，硬连线控制方式中无关，可任意设置
- 其余各位按右图所示与控制信号一一对应。可以根据个人设计方案，修改信号、引脚名称。





例如：

图中S0状态（0000），为建立起相应数据通路，需要设置的控制信号有：

IRWrite=1， PCWrite=1， MemRead = 1

ALUScrcB = 01， ALUop = 00（保证实现PC+4）

则控制存储器0号单元中应存放的二进制串为

XXX0 0001 0011 0010 000X XXXX

(X表示可任意设置）

（1）在文本文档中编辑好对应二进制编码：

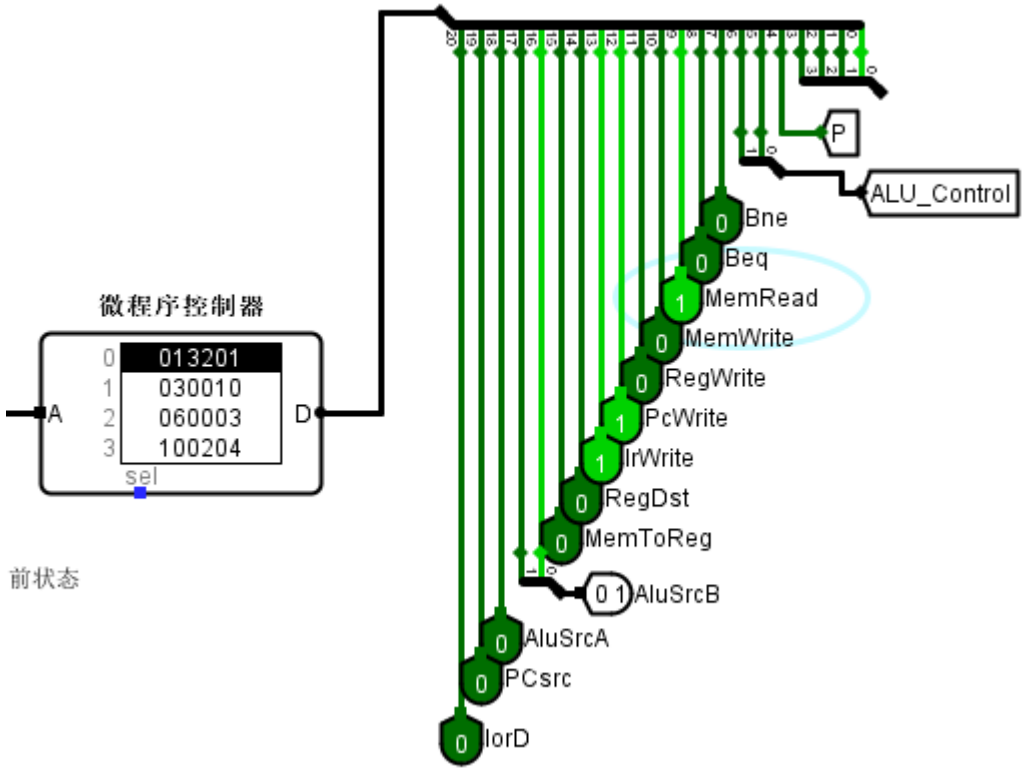
```
v2.0 raw
013201 30010
```

以“v2.0 raw” 开头；

任意两个控存单元间的内容以空格分开；

另存为.hex后缀名的文件

（2）往“微程序控制器”中载入上述编辑并保存好的文件



（3）或是利用“控制存储器微程序自动生成.xls”文件，生成微程序控制器内容

这两列可不处理

微指令功能	状态	微指令地址	IorD	PcSrc	AluSrcA	AluSrcB	MemToReg	RegDst	IrWrite	PcWrite	RegWrite	MemWrite	MemRead	BEQ	BNE	AluControl	P	下址字段	微指令	十六进制
取指令	0	0000	0	0	0	01	0	0	1	1	0	0	1	0	0	00	0	0001	00001001100100000001	13201
译码	1	0001	0	0	0	00	0	0	0	0	0	0	0	0	0	00	0	0000	00000000000000000000	0
LW1	2	0010	00	0	0	00	0	0	0	0	0	0	0	0	0	00	0	0000	00000000000000000000	0
LW2	3	0011	0	0	0	00	0	0	0	0	0	0	0	0	0	00	0	0000	00000000000000000000	0
LW3	4	0100	0	0	0	00	0	0	0	0	0	0	0	0	0	00	0	0000	00000000000000000000	0
SW1	5	0101	0	0	0	00	0	0	0	0	0	0	0	0	0	00	0	0000	00000000000000000000	0
SW2	6	0110	0	0	0	00	0	0	0	0	0	0	0	0	0	00	0	0000	00000000000000000000	0
	7	0111	0	0	0	00	0	0	0	0	0	0	0	0	0	00	0	0000	00000000000000000000	0
	8	1000	0	0	0	00	0	0	0	0	0	0				00	0	0000	00000000000000000000	0
	9	1001	0	0	0	00	0	0	0	0	0	0				00	0	0000	00000000000000000000	0
	10	1010	0	0	0	00	0	0	0	0	0	0				00	0	0000	00000000000000000000	0
	11	1011	0	0	0	00	0	0	0	0	0	0				00	0	0000	00000000000000000000	0
	12	1100	0	0	0	00	0	0	0	0	0	0				00	0	0000	00000000000000000000	0
	13	1101	1	0	0	00	0	0	0	0	0	0	0	0	0	11	0	0000	10000000000001100000	100060

控制字段
不同微指令中控制字段各位值不同，尝试根据当前微指令的功能填写各位的值，注意多位值的位宽千万不要弄错，否则后面微指令生成可能会有问题！

第1步： 在第1列安排微程序，通常取指令部分放置在0号单元，同一指令的微程序中的微指令顺序存放

第2步： 填写D到S列的微指令控制信号，注意其中aluSrcB，AluControl为2位，下地址字段4位

第3步： 完成第2步后，最后一列微指令16进制会自动更新

第3步： 将最后一列的16进制编码复制粘贴到Logisim中的控制存储器中

微指令十六进制编码直接复制粘贴到控存中

步骤（三）：电路测试

- 在存储器中载入排序程序sort.hex
- 时钟自动仿真，Windows： Ctrl+k Mac: command+k 运行程序
- 程序停机后，查看数据存储器中排序情况，有符号降序排列

000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
010	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
020	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
030	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
040	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
050	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
060	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
070	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
080	00000006	00000005	00000004	00000003	00000002	00000001	00000000	ffffffff

实验报告要求——多周期MIPS CPU设计

选做任务1

1、撰写实验报告。报告需将实验子任务的分析、设计思路与过程清晰描述，并相应贴出测试截图。

2、实验完成后，将最终的电路文件以“学号_cpu_mul.circ”命名另存。

3、在福大课程中心平台中，本门课程的“作业”中，相应提交实验报告与电路文件。