[华中科技计组实验]logisim完成8指令多周期(微程序)MIPS CPU

原创

矢遇、 已于 2022-02-24 16:09:42 修改 11059 收藏 139

分类专栏： 自己动手画cpu 文章标签： cpu

于 2020-06-03 16:26:35 首次发布

本文链接：https://blog.csdn.net/qq_44880528/article/details/106521625

自己动手画cpu 专栏收录该内容

6 篇文章 8 订阅

订阅专栏

自己动手画cpu系列 **建设中ing** *仅供参考!*

在这首推华中科技大学计算机组成原理实验课mooc连接

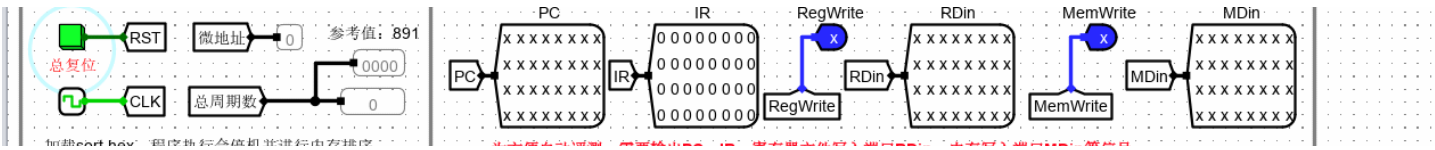
初衷：在mooc上看见了本课觉得超赞，本人已完成了课中所有的实验，在做实验的过程中有时候实验会没有思路或者有些bug会浪费很多时间，目前呢也没找到完整的答案，所以做了份自己的答案给大家困惑的时候一份参考(大神请忽视，我自己也就一弱鸡)，也就给大家卡壳的时候能有个找灵感的地方，请先独立思考，切勿抄袭。

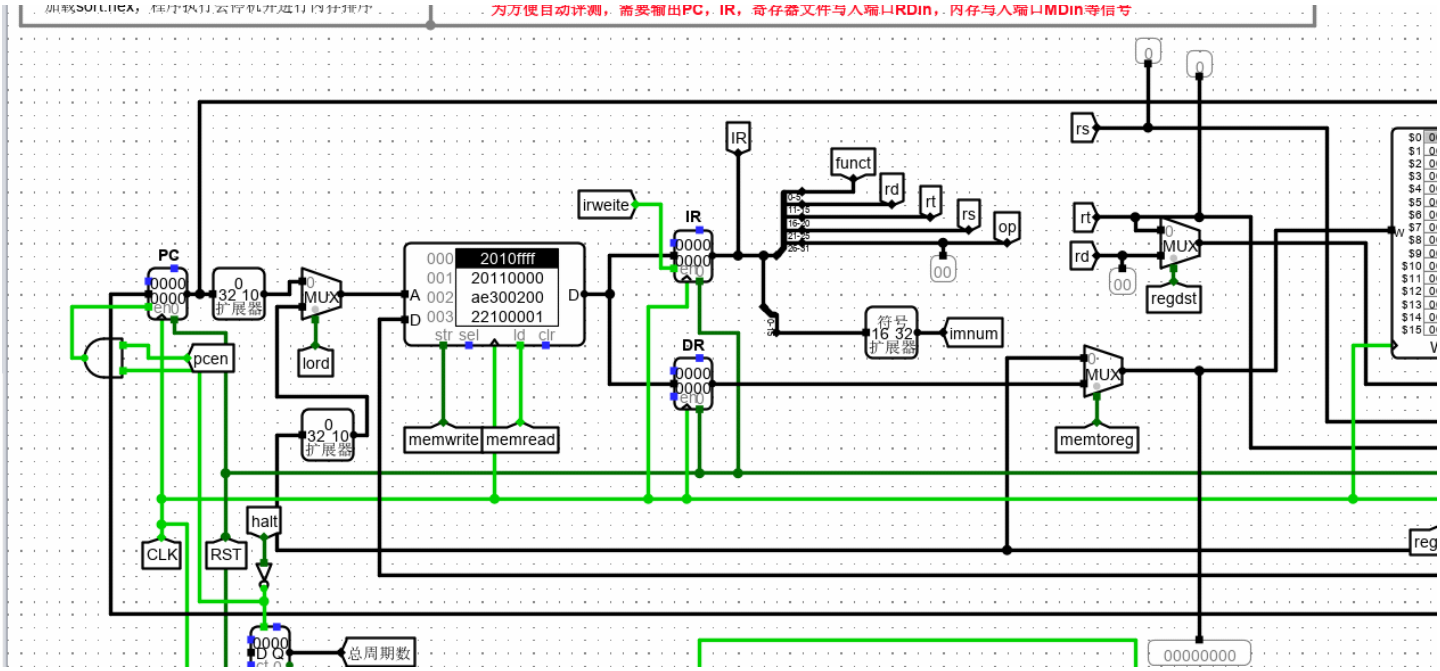
tips:每个部分都是先贴答案再写思路

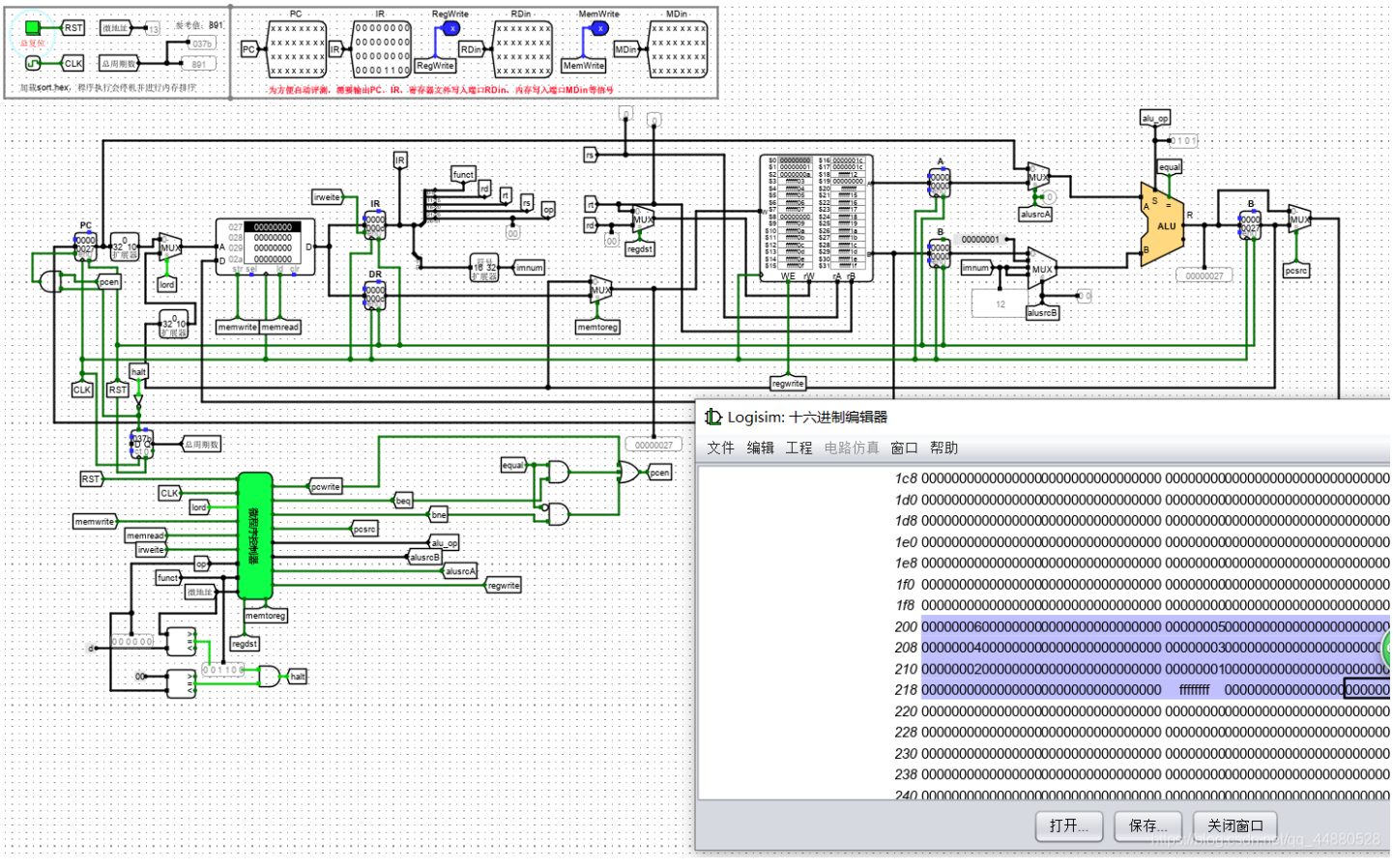
- [x] 数字逻辑基础
- [ ] 数据表示实验
- [ ] 运算器设计
- [ ] 存储器设计
- [x] MIPS CPU
  - [x] MIPS CPU必备基础知识
  - [x] 8指令单周期MIPS32 CPU
  - [x] 8指令多周期(微指令)MIPS32 CPU
  - [x] 8指令多周期(硬布线)MIPS32 CPU
  - [x] 24条指令5级流水MIPS32 CPU

已测试完降序冒泡排序

排序gif







微指令设计如下:

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
取指令	0	0000	0	0	0	01	0	0	1	1	0	0	1	0	0	00	0	0001
译码	1	0001	0	0	0	11	0	0	0	0	0	0	0	0	0	00	1	0000
LW1	2	0010	00	0	1	10	0	0	0	0	0	0	0	0	0	00	0	0011
LW2	3	0011	1	0	0	00	0	0	0	0	0	0	1	0	0	00	0	0100
LW3	4	0100	0	0	0	00	1	0	0	0	1	0	0	0	0	00	0	0000
SW1	5	0101	0	0	1	10	0	0	0	0	0	0	0	0	0	00	0	0110
SW2	6	0110	1	0	0	00	0	0	0	0	0	1	0	0	0	00	0	0000
R1	7	0111	0	0	1	00	0	0	0	0	0	0	0	0	0	10	0	1000

R2	8	1000	0	0	0	00	0	1	0	0	1	0	0	0	0	00	0	0000
BEQ	9	1001	0	1	1	00	0	0	0	0	0	0	0	1	0	00	0	0000
BNE	10	1010	0	1	1	00	0	0	0	0	0	0	0	0	1	00	0	0000
ADDI1	11	1011	0	0	1	10	0	0	0	0	0	0	0	0	0	00	0	1100
ADDI2	12	1100	0	0	0	00	0	0	0	0	1	0	0	0	0	00	0	0000
SYSCALL	13	1101	1	0	0	00	0	0	0	0	0	0	0	0	0	11	0	1101

微指令地址转移逻辑如下:

机器指令译码信号							微程序入口地址				
R_Type	ADDI	LW	SW	BEQ	BNE	SYSCALL	入口地址10进制	S3	S2	S1	S0
1						0	7	0	1	1	1
	1						11	1	0	1	1
		1					2	0	0	1	0
			1				5	0	1	0	1
				1			9	1	0	0	1
					1		10	1	0	1	0
1						1	13	1	1	0	1

有个坑爹的地方，注意syscall指令R_type和syscall都是1

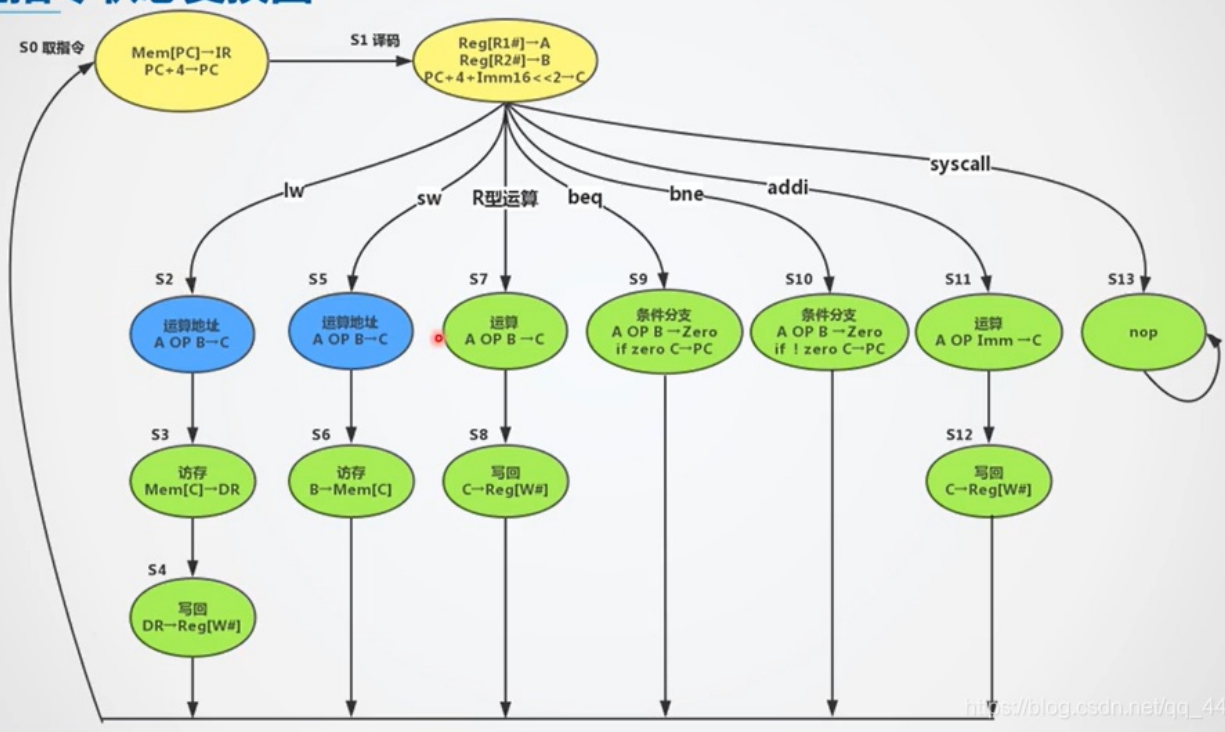
上面俩个我都测试过了仅供参考吧，设计完微地址和转移逻辑，cpu的连线还算是比较简单吧。

以下思路：

控制信号功能说明 （8条核心指令集）

#	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

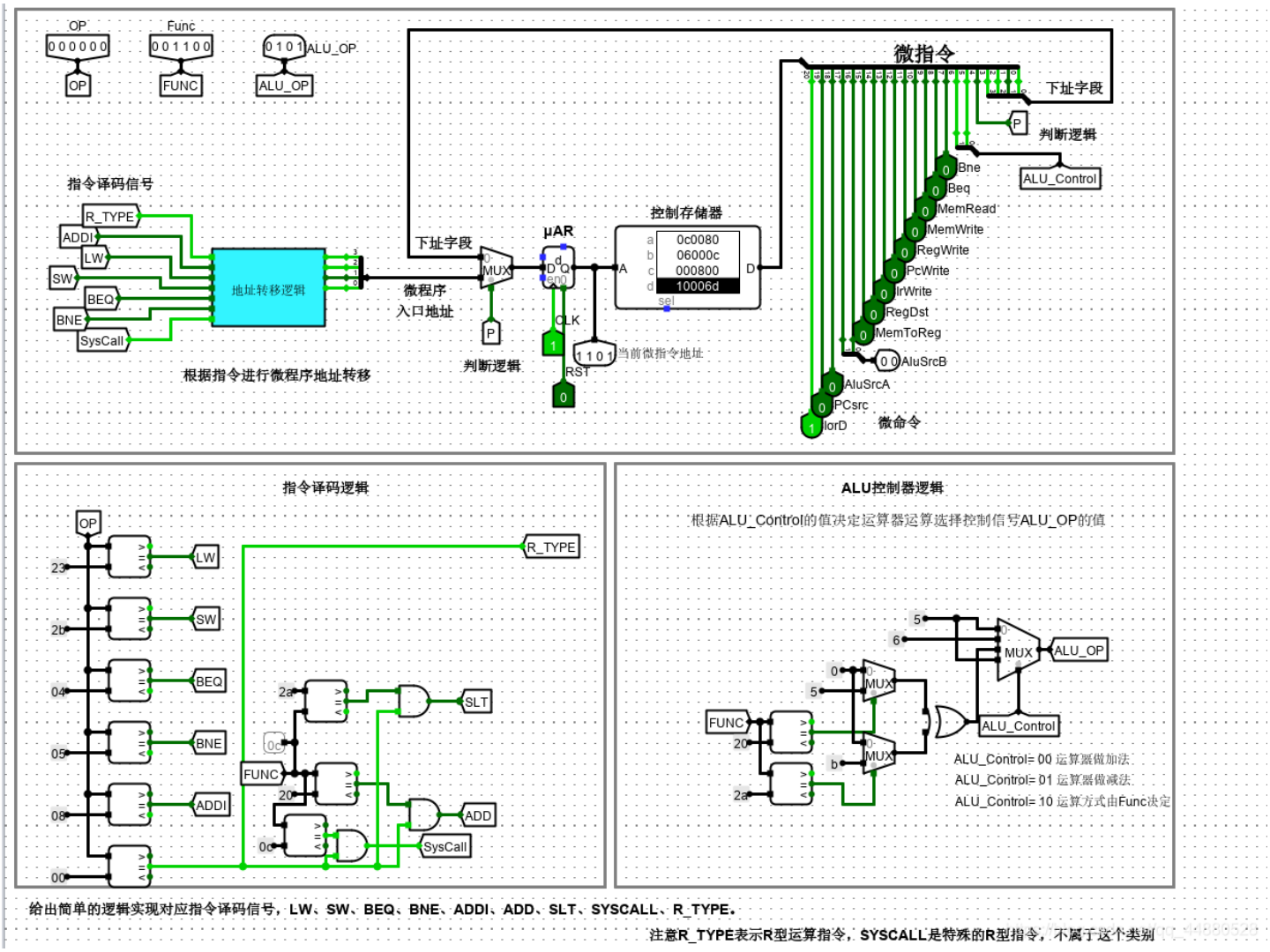
构建指令状态变换图



首先先把大体结构搭好，弄清楚每个控制信号的意义，最后再写控制器，主要说的是控制器的编写。
有了转移图，按照转移图的顺序写每个阶段的微指令

```
|_________|_________|_________|_________|_________|
|<- 取码 -> |<- 译码 -> |<- 执行 -> |<- 访存 -> |<- 写回 -> |
```

微指令中P和下址字段说明：



我感觉这设计的属实精妙，P只在译码阶段置为1，下址字段代表着下一阶段的微指令，就拿LW举例说明，初始微指令存储器地址为0000，微指令P为0下址字段为0001代表取码，首先第一个时钟到来，从存储器取出指令，控制器按照微指令下址字段，微存储器地址变为0001，第二个时钟到来，按0001地址取第二个微指令即译码，这时P为1下址字段0000(随便)，P为1二路选择器取地址转移逻辑(其实就是个编码器)的输出，因为本例是LW所以现在微指令储存器的地址为0002，第三个时钟到来，微指令为LW1即执行阶段算出要取数的地址(rs+最后16位立即数)，第四个时钟到来，微指令为LW2即访存阶段按照LW1算出的地址取出数据放入DR数据寄存器中，第五个时钟到来，微指令为LW3即写回阶段写到rt所指向的寄存器中，这时候下址字段为0000，然后第六个时钟周期微指令存储器从0000取出下一条微指令，然后重复过程。

注意：本实验停机信号不要停时钟而是停pc使能端，还有真值表一定要认真仔细填好，我感觉真值表是易错点了，本实验老师讲的跳下一条指令是pc+=4但是logisim中的存储器是按字存取的吧，pc+=1应该是正确的。其他的知识点，mooc中都有讲到，这里不再赘述了。