华中科技大学

课程实验报告

课程名称： 计算机组成原理

专业班级： **CS1902**

学　号： **I201920024**

姓　名： 木林

指导教师： 谭志虎

报告日期： **2021** 年 **12** 月 **5** 日

计算机科学与技术学院

目录

实验 CPU 设计要求

1 设计要求

1.1 单总线结构现代时序 CPU

本实验要做的包括：

1. 指令译码器
2. 微程序入口查找逻辑
3. 微程序条件判别测试逻辑
4. 微程序地址转移逻辑
5. 微程序指令存储器
6. 时序产生器（硬币线有限状态机）
7. 硬布线控制信号产生器

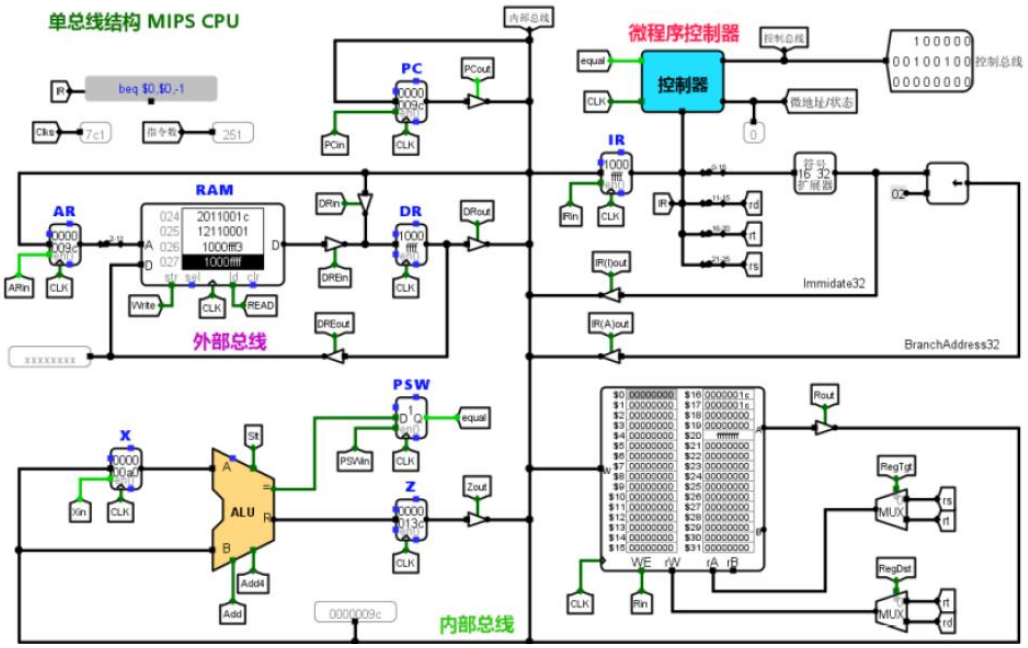


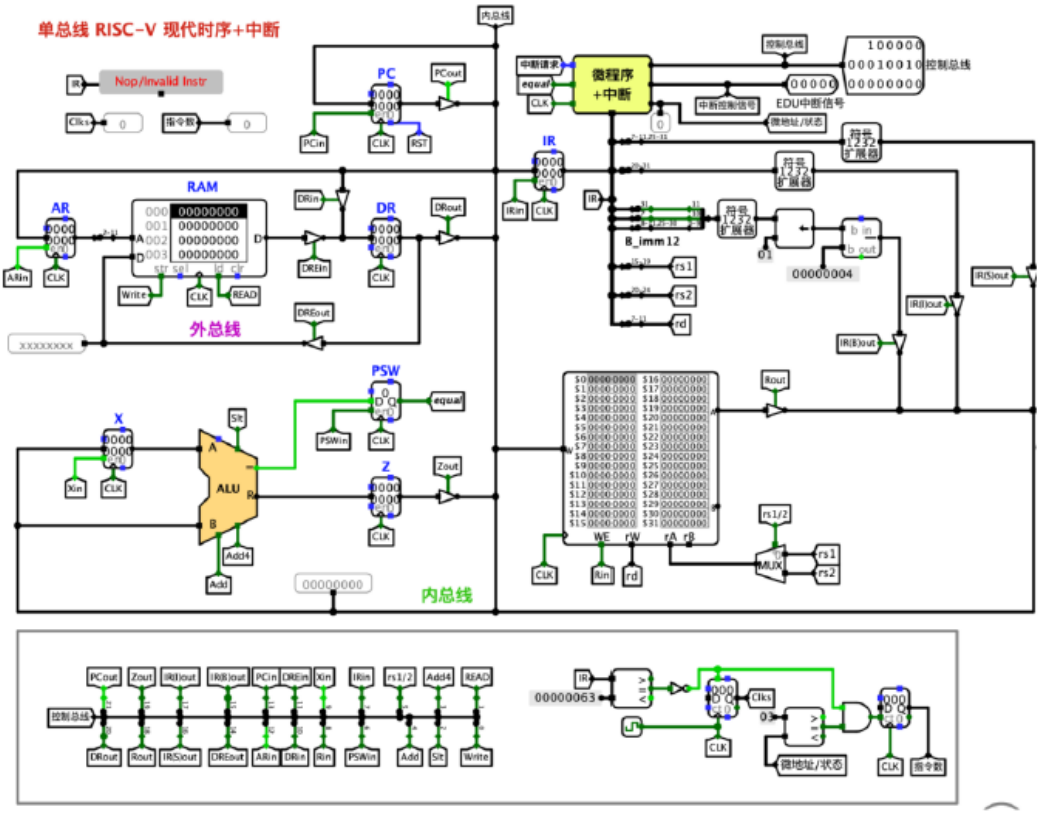
图 1.1 单总线结构现代时序总体结构图

1.2 单总线结构现代时序 CPU（带中断操作）

本实验要做的：

1. 指令译码器（带 ERET 指令）

2. 微程序入口查找逻辑
3. 微程序条件判别测试逻辑
4. 微程序地址转移逻辑
5. 微程序指令存储
6. 时序产生器（硬币线有限状态机）
7. 硬布线控制信号产生器



1.3 单总线结构三级时序变长指令 CPU

本实验要做的：

1. 指令译码器
2. 硬币线时序发生器
3. 硬币线时序发生器输出逻辑
4. 硬币线控制器组合逻辑单位
5. 硬币线控制器

图 1.2 单总线结构现代时序（带中断）总体结构图

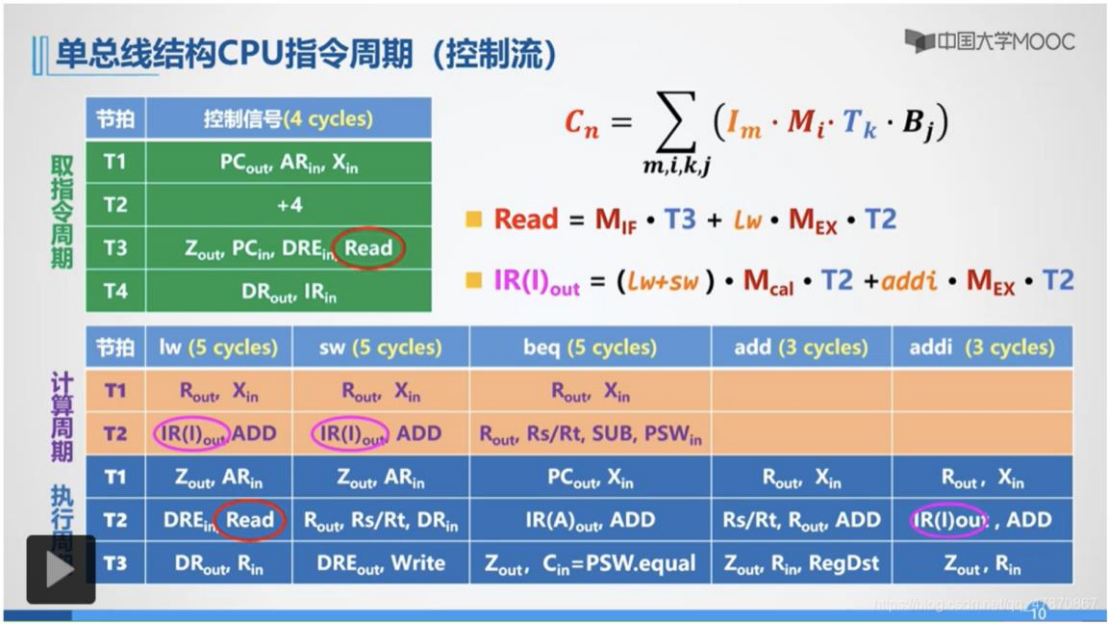
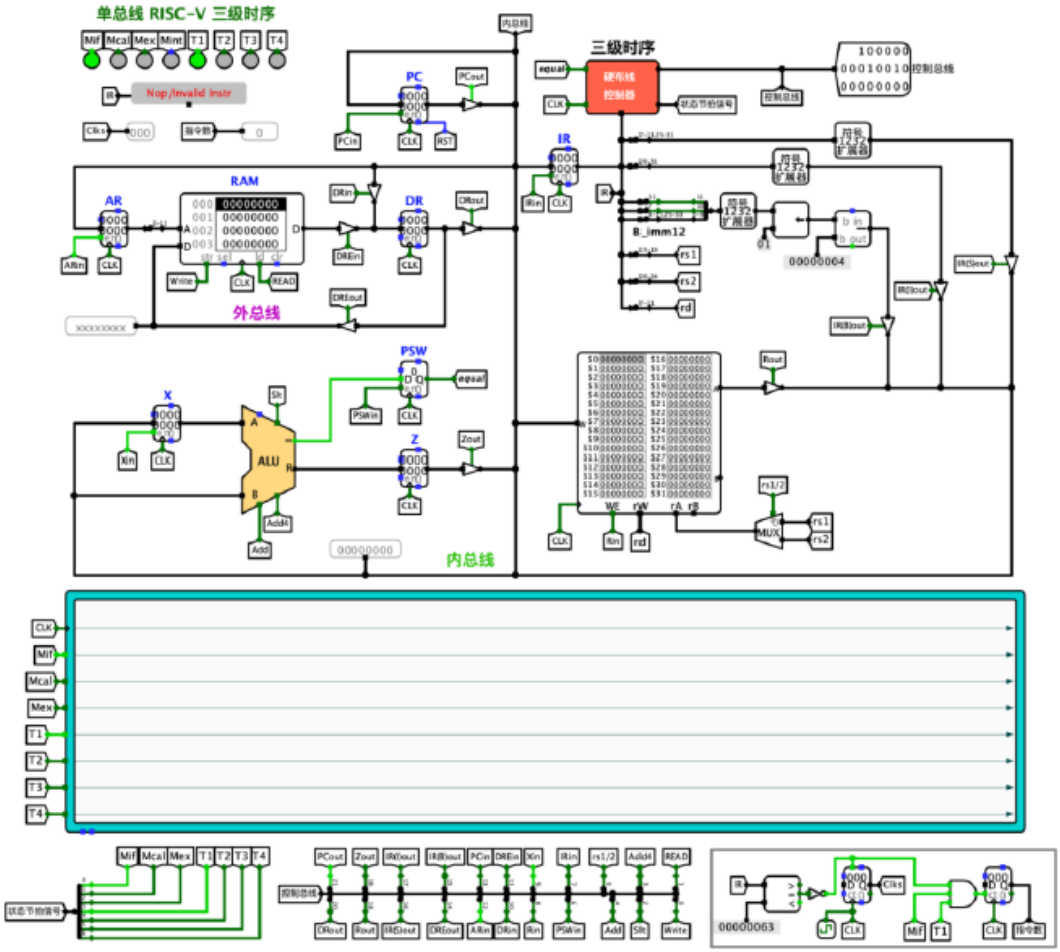


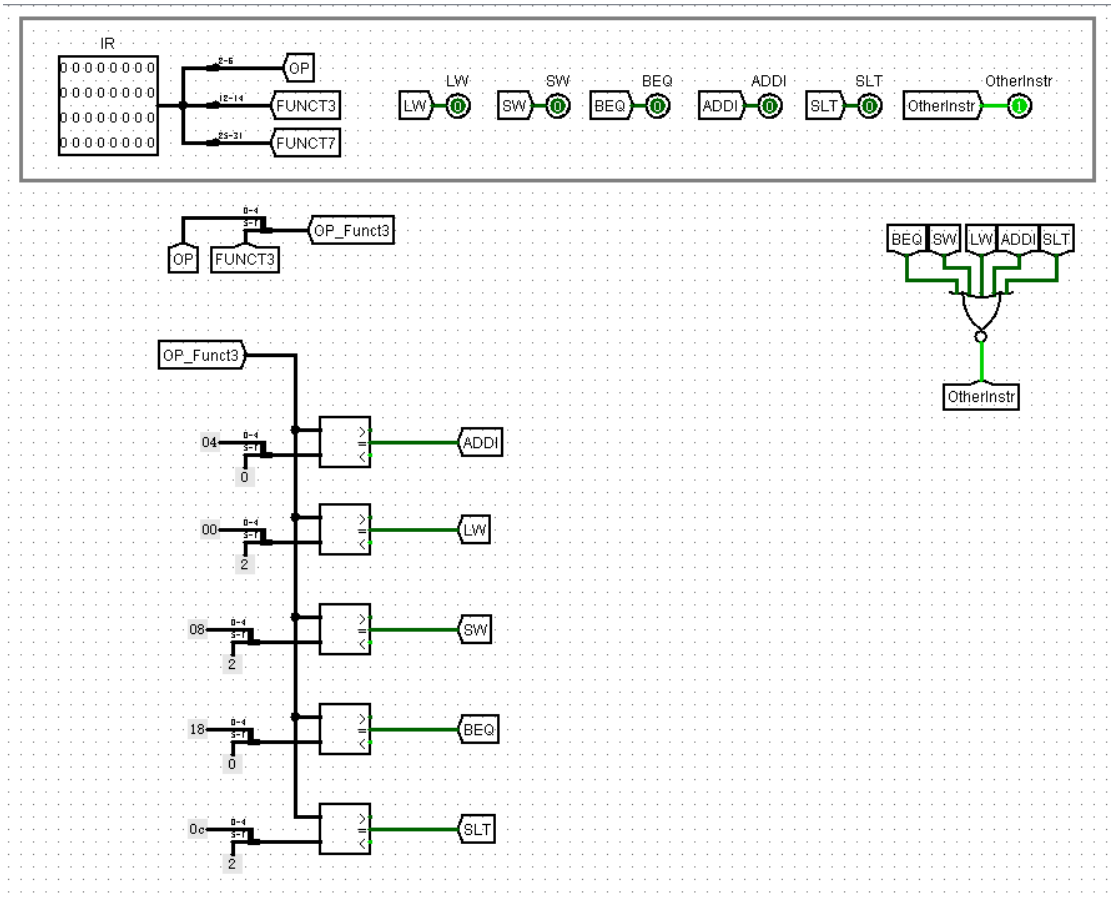
图 1.3a 中国大学 mooc 中的单总线结构三级时序变长指令 CPU 指令期



2.1 单总线结构现代时序 CPU

- 指令译码器

图 1.4 单总线结构三级时序变长指令总体结构图



输入：32 位指令 IR

输出：指令译码器的结果：LW 信号代表指令是否为 LW 指令，SW 信号表表指令是否为 SW 指令，BEQ 信号代表指令是否为 BEQ 指令，ADDI 信号代表指令是否为 ADDI 指令，SLT 信号代表指令是否为 SLT 指令

实现逻辑：将指令 IR 的 2—6（opcode）和 12-14（func3）位提取出来进行比对，用 5 个比较器来判断该指令的 opcode 和 func3 是否与标准指令匹配。

图 1.5 单总线结构现代时序指令译码器总体结构图

图 1.5 单总线结构现代时序指令译码器总体结构

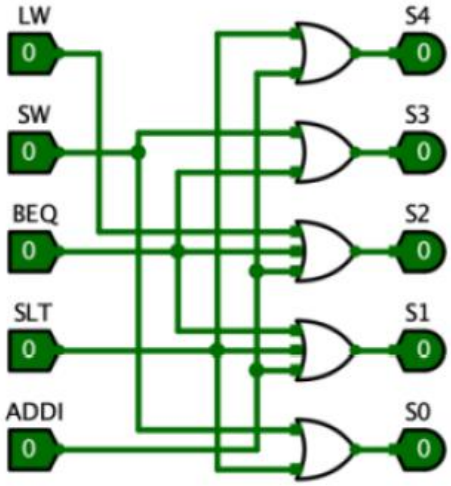
- 

图 1.6 单总线结构现代时序微程序入口查找逻辑总体结构图

输入：5 个指令选择信号，代表译码出来的指令类型。

输出：微程序入口地址（$S=[S_4S_3S_2S_1S_0]$）

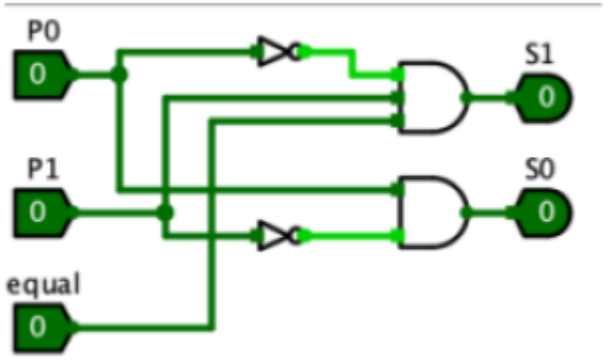
实现逻辑：利用 Excel 表构造组合逻辑，然后在 logism 组合逻辑生成器生成电路。

机器指令译码信号						微程序入口地址					
LW	SW	BEQ	SLT	ADDI	ERET	入口地址 10进制	S4	S3	S2	S1	S0
1						4	0	0	1	0	0
	1					9	0	1	0	0	1
		1				14	0	1	1	1	0
			1			19	1	0	0	1	1
				1		22	1	0	1	1	0

图 1.7 单总线结构现代时序微程序入口查找逻辑组合逻辑设计表

- 微 程 序 条 件 判 别 逻 辑

图 1.8 单总线结构现代时序微程序条件判别逻辑组合总体结构图



实现逻辑：微程序流程控制信号 P=[P_1P_0]和表示相等关系信号 equal。
微程序地址转移控制信号(S=[S_1S_0]):S=0 代表是下一条地址，S=1 代表取入口地址，S=2 代表是取 BEQ 指令且当前运算是相等的跳转地址。利用 Excel 表构造组合逻辑，然后在 logism 生成电路。

微指令功能		PCout	DRout	Zout	Rout	PCin	ARin	DREin	DRin	Xin	Rin	IRin	rs1/2	Add	SIt	READ	WRITE	P0	P1	P2	下址DEC	微指令	微指令十六进制
取指令	0	1					1			1											1	00000000010010000000000000000	20120001
取指令	1														1						2	00000000000000000010000000001	802
取指令	2			1			1		1								1				3	00100000101000000000100000011	8280203
取指令	3		1								1							1			0	01000000000001000000010000000	10008080
LW	4				1					1											5	00010000000010000000000000010	4020005
LW	5					1								1							6	00001000000000001000000000110	2001006
LW	6			1			1														7	00100000010000000000000000011	8100007
LW	7							1								1					8	00000000001000000001000001000	80208
LW	8		1								1										0	01000000000001000000000000000	10010000
SW	9				1					1											10	00010000000010000000000000101	402000A
SW	10					1								1							11	00000100000000000100000000101	100100B
SW	11			1			1														12	00100000010000000000000000110	810000C
SW	12				1				1					1							13	00010000000100001000000000111	404200D
SW	13					1											1				0	00000001000000000000010000000	400100
BEQ	14				1					1											15	00010000000010000000000000111	402000F
BEQ	15				1								1	1						1	0	00100000000000110000001000000	4006040
BEQ	16	1								1											17	00000000001000000000001000	20020011
BEQ	17					1								1							18	00000010000000001000000010010	801012
BEQ	18			1				1													0	01000001000000000000000000000	8200000
SLT	19				1					1											20	00010000000010000000000010100	4020014
SLT	20				1								1		1						21	00010000000000001001000010101	4002415
SLT	21			1							1										0	01000000000001000000000000000	8010000
ADDI	22				1					1											23	00010000000010000000000010111	4020017
ADDI	23					1								1							24	00001000000000001000000011000	2001018
ADDI	24			1							1										0	01000000000001000000000000000	8010000

- 控制寄存器

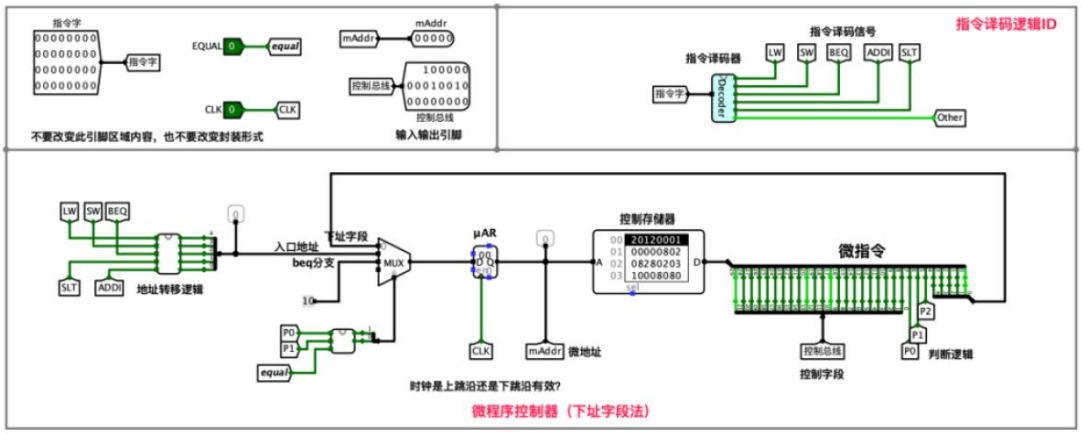
实现逻辑：微程序的地址对应硬布线的时序，我们只需要对照着硬布线的控制信号输出表填写控制信号，然后根据 FSM 的关系设置下一条微程序的地址，设计逻辑如下：

1. 在不需要跳转的微程序处，下一条微程序是本指令地址+1
2. 在取指令的最后一条微程序标记下一条微程序的地址是微程序入口生成地址逻辑结果。

图 1.9 单总线结构现代时序微程序控制寄存器内容

3. 在每一条指令的最后一条微程序标记下，一条指令是取指令的第一条指令
4. 对于 equal 的判断：如果在 S15 条微程序的最后设置下一条指令与 equal 信号有关，交由微程序条件判别逻辑来判断，如果是相等的话(equal=1)，跳转到 S16。如果不是，返回取指令的第一条指令。

- 微程序控制器



实现逻辑：其他功能模块已经在之前说明，这里最重要的就是下一条微程序的地址：需要根据微程序条件判别逻辑的结果来进行多路选择。如果是下一条地址就对应当前微程序的下地址字段。如果是入口地址就接入入口地址查找逻辑的输出。如果是 beq 分支就跳转到 0x10。

- 硬布线时许产生器

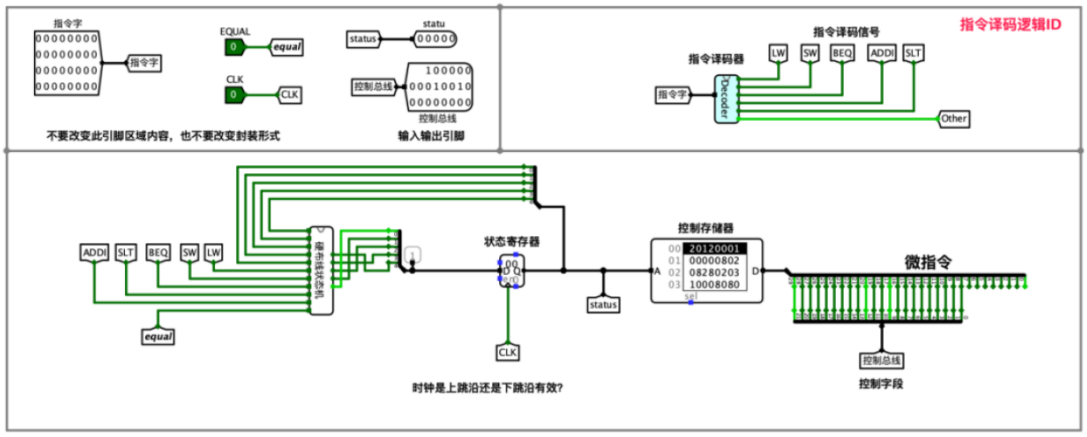
实现逻辑：Excel 生成组合逻辑，然后在 logism 生成电路

图 1.10 单总线结构现代时序微程序控制器总体结构图

- 硬布线控制器

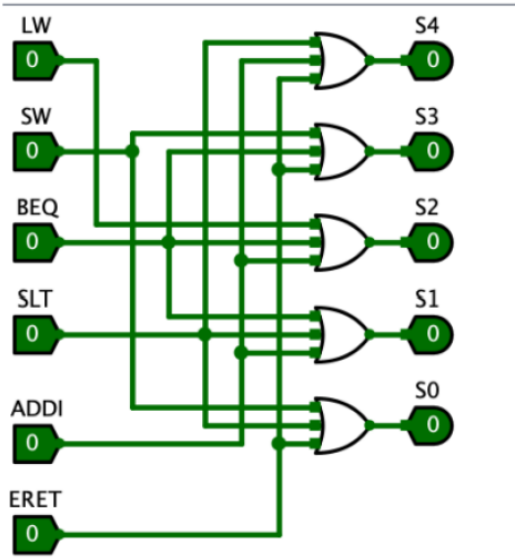
实现逻辑：主要的就是状态寄存器右边部分是当前状态，左边部分是次

态，将当前状态和指令译码器的输出作为 FSM 的输入连线即可。因为硬布线的状态和微程序的地址是一一对应的，这个时候可以借用之前做的微程序控制存储器来读取控制总线输出。



2.2 单总线结构现代时序 **CPU**（带中断操作）

- 指令译码器
  与 2.1 的相同。略
- 微程序入口查找逻辑



1 图 1.2 单总线结构现代时序（带中断）微程序入口查找逻辑总体结构图

输入：6 个指令选择信号，代表译码出来的指令类型。

输出：微程序入口地址（$S=[S_4S_3S_2S_1S_0]$）

实现逻辑：用 Excel 构造组合逻辑，然后在 logism 生成电路

- 微程序条件判别逻辑

机器指令译码信号						微程序入口地址					
LW	SW	BEQ	SLT	ADDI	RET	入口地址10进制	S4	S3	S2	S1	S0
1						4	0	0	1	0	0
	1					9	0	1	0	0	1
		1				14	0	1	1	1	0
			1			19	1	0	0	1	1
				1		22	1	0	1	1	0
					1	25	1	1	0	0	1

图 1.13 单总线结构现代时序（带中断）微程序入口查找逻辑组合设计表

实现逻辑：用 Excel 表构造组合逻辑，然后在 logism 生成电路。

输入 （填1或0，不填为无关项x）							
P0	P1	P2	equal	IntR	S2	S1	S0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	1	0	0	1	0
0	1	1	0	1	0	1	1
0	1	1	0	0	1	0	0
0	0	1	0	1	0	1	1
0	0	1	1	1	0	1	1
0	0	1	0	0	1	0	0
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	0
0	1	0	0	1	1	0	0

图 1.14 单总线结构现代时序（带中断）微程序条件判别逻辑组合逻辑设计表

- 控制寄存器

实现逻辑：微程序的地址对应硬布线的时序，我们只需要对照着硬布线的控制信号输出表填写控制信号，然后根据 FSM 关系设计流程控制信号 P。

1. 当这个是取地址指令的最后一条微程序的时候，这个时候标记 P_0 要找入

口查找逻辑寻找入口地址。

2. 当这个是判断 equal 分支的时候，这个时候标记 P_1 有可能要跳转到处理 equal 分支的微程序
3. 当这个是每一条指令的最后一条微程序的时候，标记 P_2 代表结束。

微指令功能		PCout	DRout	Zout	Rout	IR0out	IR3out	IR5out	DREout	PCin	ARin	DREin	DRin	Xin	Rin	IRin	PSWin	rs1/2	Add	Add4	Slt	READ	WRITE	EPCout	EPCin	MMrd	STI	CLI	P0	P1	P2	微指令十六进制
取指令	0	1									1			1																		20120000
取指令	1																			1												800
取指令	2			1						1		1										1										8280200
取指令	3		1													1													1			10008004
LW	4				1									1																		4020000
LW	5					1													1													2001000
LW	6			1							1																					8100000
LW	7											1										1										80200
LW	8		1												1																1	10010001
SW	9				1									1																		4020000
SW	10						1												1													1001000
SW	11			1							1																					8100000
SW	12				1								1					1														4042000
SW	13							1															1								1	400101
BEQ	14				1									1																		4020000
BEQ	15				1												1	1												1	1	4006003
BEQ	16	1												1																		20020000
BEQ	17							1											1													801000
BEQ	18			1						1																					1	8200001
SLT	19				1									1																		4020000
SLT	20				1													1			1											4002400
SLT	21			1											1																1	8010001
ADDI	22				1									1																		4020000
ADDI	23					1													1													2001000
ADDI	24			1											1																1	8010001
ERET	25									1															1			1			1	200091
中断响应	26	1																								1			1			20000048
中断响应	27									1																	1				1	200021
																																正确

- 微程序控制器

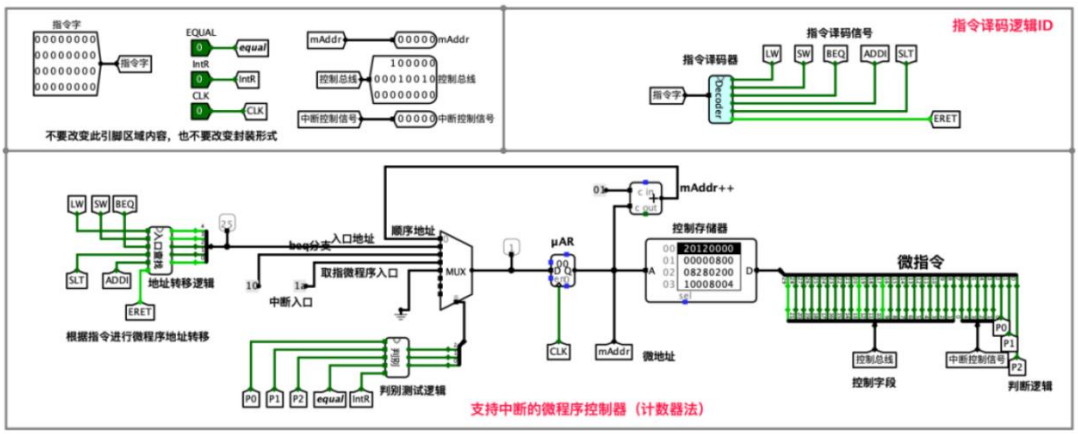


图 1.16 单总线结构现代时序（带中断）微程序控制器总体结构图

实现逻辑：其他功能模块已经在之前说明，这里最重要的就是下一条微程序的地址：需要根据微程序条件判断逻辑的结果来进行多路选择

图 1.15 单总线结构现代时序（带中断）微程序控制寄存器内容

- 硬币线时序产生器

实现逻辑：用 Excel 构造组合逻辑，然后在 logism 生成电路。

当前状态(现态)						输入信号								下一状态（次态）					
S4	S3	S2	S1	S0	现态10进制	LW	SW	BEQ	SLT	ADDI	URET	IR	EQUAL	次态10进制	N4	N3	N2	N1	N0
0	0	0	1	1	3		1							9	0	1	0	0	1
0	0	0	1	1	3			1						14	0	1	1	1	0
0	0	0	1	1	3				1					19	1	0	0	1	1
0	0	0	1	1	3					1				22	1	0	1	1	0
0	0	1	0	0	4									5	0	0	1	0	1
0	0	1	0	1	5									6	0	0	1	1	0
0	0	1	1	0	6									7	0	0	1	1	1
0	0	1	1	1	7									8	0	1	0	0	0
0	1	0	0	0	8							0		0	0	0	0	0	0
0	1	0	0	1	9									10	0	1	0	1	0
0	1	0	1	0	10									11	0	1	0	1	1
0	1	0	1	1	11									12	0	1	1	0	0
0	1	1	0	0	12									13	0	1	1	0	1
0	1	1	0	1	13							0		0	0	0	0	0	0
0	1	1	1	0	14									15	0	1	1	1	1
0	1	1	1	1	15								1	16	1	0	0	0	0
0	1	1	1	1	15							0	0	0	0	0	0	0	0
1	0	0	0	0	16									17	1	0	0	0	1
1	0	0	0	1	17									18	1	0	0	1	0
1	0	0	1	0	18							0	1	0	0	0	0	0	0
1	0	0	1	1	19									20	1	0	1	0	0
1	0	1	0	0	20									21	1	0	1	0	1
1	0	1	0	1	21							0		0	0	0	0	0	0
1	0	1	1	0	22									23	1	0	1	1	1
1	0	1	1	1	23									24	1	1	0	0	0
1	1	0	0	0	24							0		0	0	0	0	0	0
0	0	0	1	1	3						1			25	1	1	0	0	1
1	1	0	0	1	25							0		0	0	0	0	0	0
1	1	0	0	1	25							1		26	1	1	0	1	0
0	1	0	0	0	8							1		26	1	1	0	1	0
0	1	1	0	1	13							1		26	1	1	0	1	0
0	1	1	1	1	15							1	0	26	1	1	0	1	0
1	0	0	1	0	18							1		26	1	1	0	1	0
1	0	1	0	1	21							1		26	1	1	0	1	0
1	1	0	0	0	24							1		26	1	1	0	1	0
1	1	0	0	1	25							1		26	1	1	0	1	0
1	1	0	1	0	26									27	1	1	0	1	1
1	1	0	1	1	27									0	0	0	0	0	0

- 硬布线控制器

输入：指令字，EQUAL 和时钟

输出：控制总线输出。

实现逻辑：主要就是状态寄存器右边部分是当前状态，左边部分是次态，将当前状态和指令译码器的输出作为 FSM 的输入连线即可。因为硬布线的状态和微程序的地址是一一对应的，这个时候可以借用之前做的微程序控制寄存器来读取控制总线输出。

图 1.17 单总线结构现代时序（带中断）硬布线时序产生器组合逻辑设计表

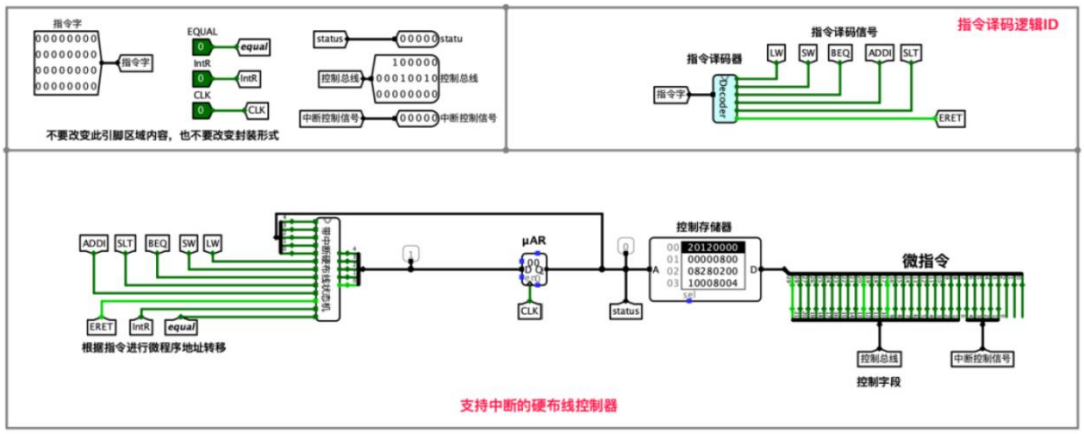


图 1.18 单总线结构现代时序硬布线控制器（带中断）总体结构

- 中断控制器件

IE 寄存器：关中断的时候异步置为 1，开中断的时候异步置为 0，输出为寄存器保存内容相反的部分。

mEPC 寄存器：和数据通路的其他锁存器是一种构造，就是在写入问好为 0 的时候，通过使能端控制寄存器忽略时钟信号，不把输入端内容寄存下来，锁存器输出接三态门，三态门只有在 EPC 输出信号有效的情况下，才能进行输出，不能进行输出的时候三态门阻挡。

中断地址查找：根据中断控制的中断类型输出进行分支判断，分支判断使用多路选择器，对于输出类型为 0&3 的就不与处理，对于分支 1，接上代表 1 号中断的中断入口地址的常量，对于分支 2 接上对于 2 号中断的中断入口地址常量。

对于中断地址的访问，首先查看这个不带中断的程序，发现程序在 00000063 停止，第 42 行是正常程序的最后一条；对于带中断的程序，从主程序在第 42 行结束，从第 43 行开始就是中断程序，接着找到中断程序的第一条指令 00810113，在第 43 行，我们接着往下找，发现在第 61 行也发现了指令 00810113，推测这是第二个中断程序的第一条指令，那么中断入口指令分别是第 41 条指令和第 59 条指令；又已知一条指令占 4 个字节，所以说我们可以知道第一个中断指令的地址就是$[41*4]_{10}$和$[59*4]_{10}$。我们可以使用 Rars 来分析汇编语言中各个标签的地址,也可以得到答案。

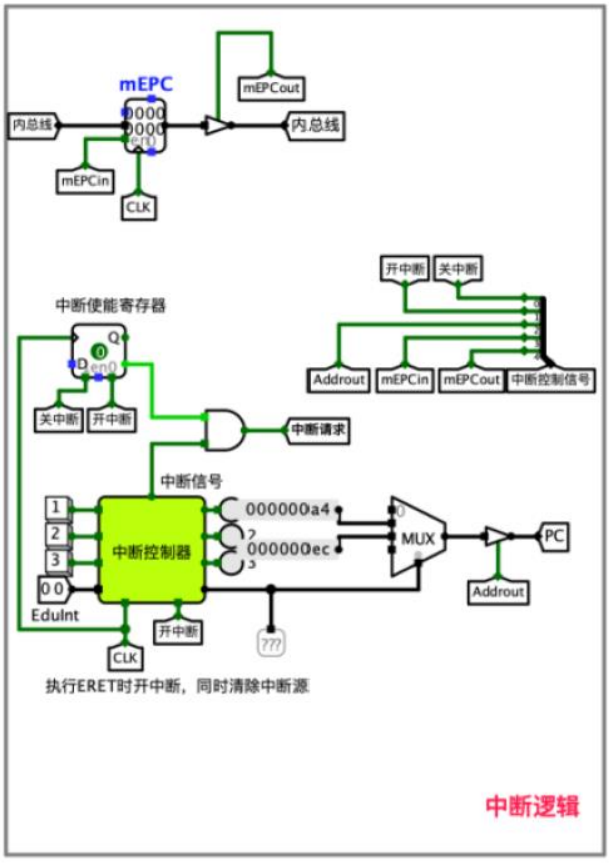


图 1.19 总线结构现代时序（带中断）中断逻辑总体结构

2.3 单总线结构三级时序变长指令 CPU

- 指令译码器

- 硬布线时序发生器有限状态机

实现逻辑：利用 Excel 表构造组合逻辑，然后在 logism 生成电路

当前状态(现态)					输入信号							下一状态（次态）				
S3	S2	S1	S0	现态10进制	LW	SW	BEQ	SLT	ADDI	ERET	IntR	次态10进制	N3	N2	N1	N0
0	0	0	0	0								1	0	0	0	1
0	0	0	1	1								2	0	0	1	0
0	0	1	0	2								3	0	0	1	1
0	0	1	1	3	1							4	0	1	0	0
0	0	1	1	3		1						4	0	1	0	0
0	0	1	1	3			1					4	0	1	0	0
0	1	0	0	4								5	0	1	0	1
0	1	0	1	5								6	0	1	1	0
0	0	1	1	3				1				6	0	1	1	0
0	0	1	1	3					1			6	0	1	1	0
0	1	1	0	6								7	0	1	1	1
0	1	1	1	7								8	1	0	0	0
1	0	0	0	8								0	0	0	0	0
0	0	1	1	3	0	0	0	0	0			6	0	1	1	0

图 1.20 单总线结构三级时序硬布线时序发生器 FSM 组合逻辑设计表

- 硬布线时序发生器组合逻辑输出组件

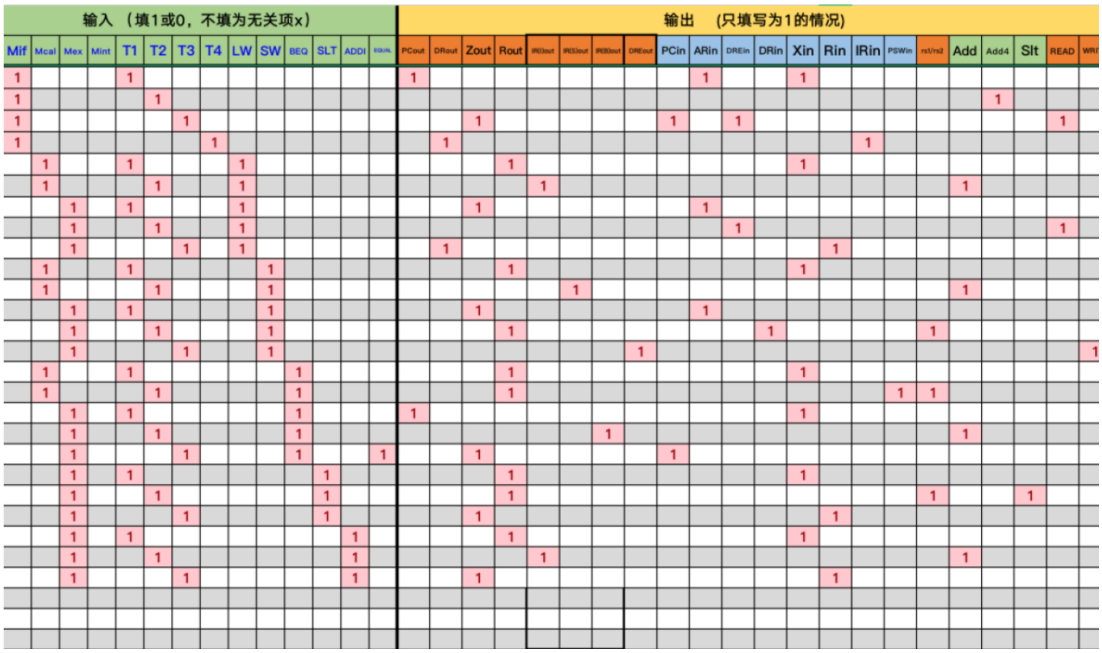
实现逻辑：用 Excel 表构造组合逻辑，然后在 logism 生成电路

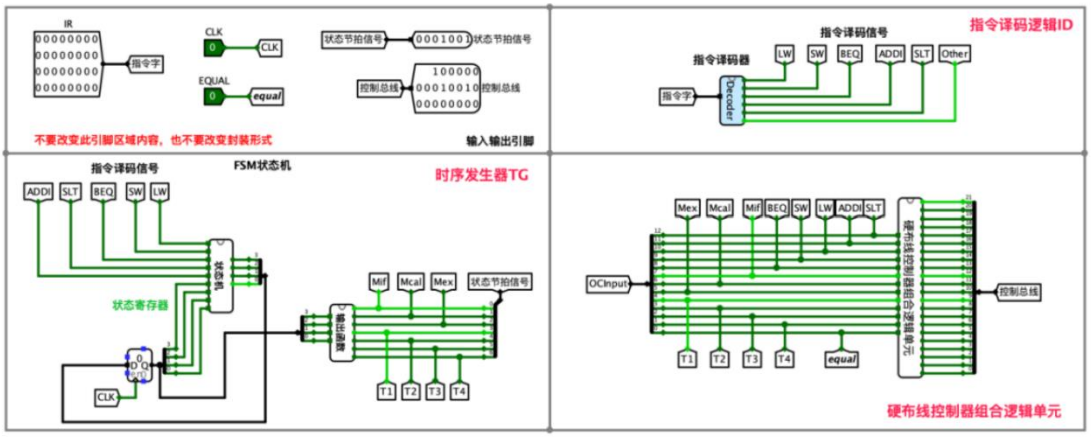
当前状态(现态)					输出							
S3	S2	S1	S0	现态10进制	Mif	Mcal	Mex	Mint	T1	T2	T3	T4
0	0	0	0	0	1				1			
0	0	0	1	1	1					1		
0	0	1	0	2	1						1	
0	0	1	1	3	1							1
0	1	0	0	4		1			1			
0	1	0	1	5		1				1		
0	1	1	0	6			1		1			
0	1	1	1	7			1			1		
1	0	0	0	8			1				1	

- 硬布线组合逻辑输出组件

实现逻辑：用 Excel 表构造组合逻辑，然后在 logism 生成电路。

图 1.21 单总线结构三级时序硬布线时序发生器输出函数逻辑设计表





硬布线控制器

实现逻辑：主要的就是状态寄存器右边部分是当前状态，左边部分是次态，将当前状态和指令译码器的输出作为 FSM 的输入连线即可。根据当前状态序号，可以通过硬布线组合逻辑输出器件输出各种控制信号的输出。

图 1.22 单总线结构三级时序硬布线组合逻辑输出函数逻辑设计表