

# 华中科技大学

## 2020

### 计算机组成原理

### · 实验报告 ·

专    业：        计算机科学与技术

班    级：        CS1806

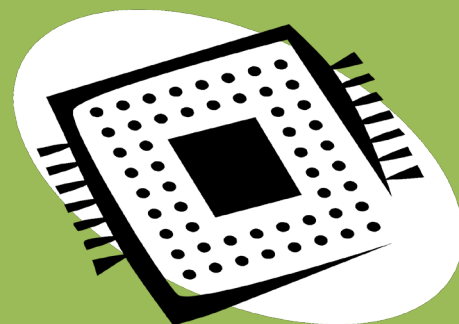
学    号：        U201813676

姓    名：        刘汉鹏

电    话：        18372637700

邮    件：        1359947339@qq.com

完成日期：        2020-12-10



计算机科学与技术学院

# 华中科技大学课程实验报告

---

## 目 录

1	CPU 设计实验 .....	2
1.1	设计要求 .....	2
1.2	方案设计 .....	2
1.3	实验步骤 .....	5
1.4	故障与调试 .....	14
1.5	测试与分析 .....	15
2	总结与心得 .....	18
2.1	实验总结 .....	18
2.2	实验心得 .....	18
	参考文献 .....	20

## 1 CPU 设计实验

### 1.1 设计要求

构建一个 32 位 MIPS CPU 处理器，包括单周期硬布线 CPU、多周期微程序 CPU 以及多周期硬布线 CPU，该处理器应支持核心指令集中列出的所有指令，见表 1.1。具体指令功能参见附件中的 MIPS 标准文档。最终设计完成的 CPU 应能运行标准测试程序。

表 1.1 核心指令集

#	指令	格式	备注
1	Add Immediate	addi \$rt, \$rs, immediate	指令功能及指令格式 参考 MIPS32 指令集
2	Load Word	lw \$rt, offset(\$rs)	
3	Store Word	sw \$rt, offset(\$rs)	
4	Branch on Equal	beq \$rs, \$rt, label	
5	Set Less Than	slt \$rd, \$rs, \$rt	
6	OtherInstr		其他指令

### 1.2 方案设计

#### 1.2.1 单总线 CPU 设计（定长指令周期 3 级时序）部件

- (1) MIPS 指令译码器：指令译码器是控制器核心功能部件，负责将指令字翻译成一根根的指令译码信号，每一根指令译码信号代表一条具体的指令；
- (2) 时序发生器状态机：状态机负责现态与次态的转换；
- (3) 时序发生器输出函数：输出函数为组合逻辑，输入为状态寄存器的现态输出，输出为状态周期电位和节拍电位信号；
- (4) 硬布线控制器组合逻辑单元：根据现态和指令周期，生成相应的控制信号；
- (5) 硬布线控制器：产生控制信号，控制指令执行的数据通路；
- (6) 单总线 CPU（3 级时序）：执行相应的指令程序，完成相应的功能。

# 华中科技大学课程实验报告

以上所提到的功能部件在 logisim 中的外观如下表 1-2 所示

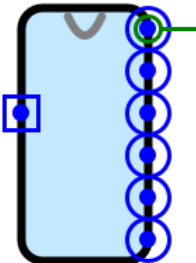
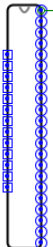
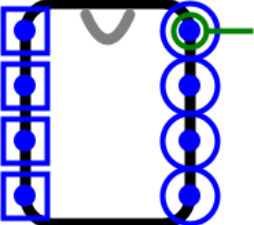
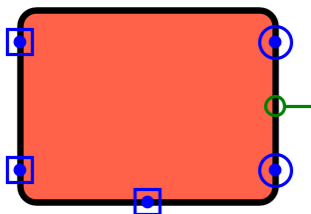
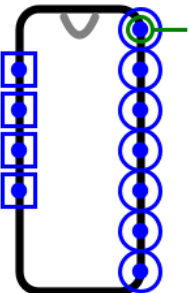
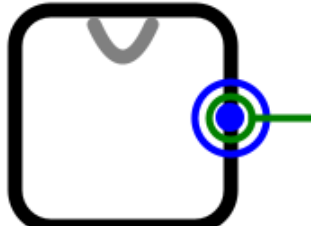
单周期 CPU 功能部件	部件外观 (logisim)	单周期 CPU 功能部件	部件外观 (logisim)
指令译码器		硬布线控制器 组合逻辑单元	
时序发生器状态机		硬布线控制器	
时序发生器输出函数		单总线 CPU (3 级时序)	

表 1-2. 单总线 CPU (3 级时序) 功能部件

## 1.2.2 单总线 CPU 设计 (现代时序) 部件

- (1) MIPS 指令译码器：指令译码器是控制器核心功能部件，负责将指令字翻译成一根根的指令译码信号，每一根指令译码信号代表一条具体的指令；
- (2) 微程序入口查找逻辑：根据指令字查找微程序的入口地址；
- (3) 条件判别测试逻辑：根据相应的判别测试位，进行微程序分支；
- (4) 微程序控制器：输入指令字和判别标志，输出相应的微指令地址和控制字段；
- (5) 硬布线状态机：输入现态和指令字，输出次态，完成相应的状态转换；

# 华中科技大学课程实验报告

- (6) 硬布线控制器：根据指令字和现态，输出相应的控制信号，控制数据通路；
- (7) 采用微程序的单总线 CPU：执行一系列的指令，完成相应的程序。

以上所提到的功能部件在 logisim 中的外观如下表 1-3 所示

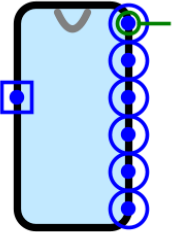
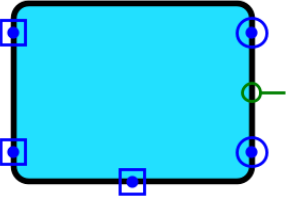
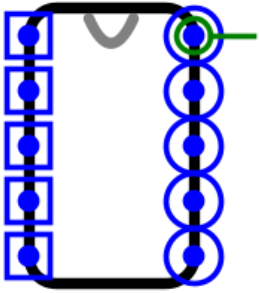
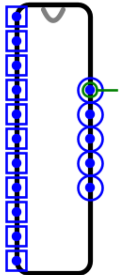
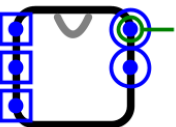
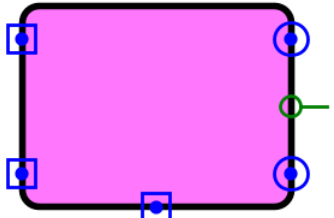
多周期 CPU 功能部件	部件外观 (logisim)	多周期 CPU 功能部件	部件外观 (logisim)
MIPS 指令译码器		微程序控制器	
微程序入口查找逻辑		硬布线状态机	
条件判别测试逻辑		硬布线控制器	

表 1-3. 单总线 CPU（现代时序）功能部件

## 1.2.3 现代时序中断机制实现部件

- (1) MIPS 指令译码器：指令译码器是控制器核心功能部件，负责将指令字翻译成一根根的指令译码信号，每一根指令译码信号代表一条具体的指令；
- (2) 微程序入口查找逻辑：根据指令字查找微程序的入口地址；
- (3) 条件判别测试逻辑：根据相应的判别测试位，进行微程序分支；
- (4) 微程序控制器：输入指令字和判别标志，输出相应的微指令地址和控制字段；

# 华中科技大学课程实验报告

- (5) 硬布线状态机：输入现态和指令字，输出次态，完成相应的状态转换；
- (6) 硬布线控制器：根据指令字和现态，输出相应的控制信号，控制数据通路；
- (7) 采用微程序的单总线 CPU：执行一系列的指令，完成相应的程序，且支持中断响应。

以上所提到的功能部件在 logisim 中的外观如下表 1-4 所示

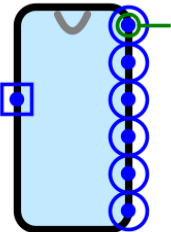
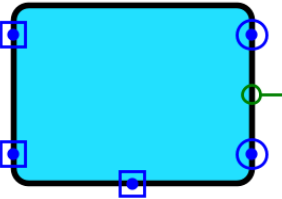
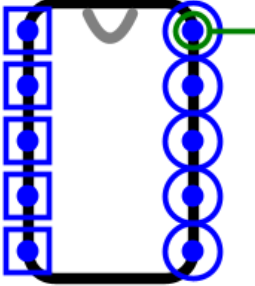
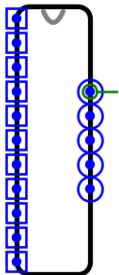
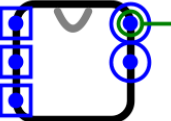
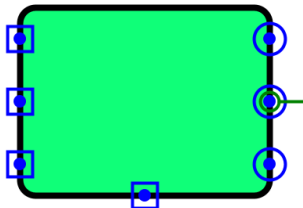
多周期 CPU 功能部件	部件外观 (logisim)	多周期 CPU 功能部件	部件外观 (logisim)
MIPS 指令译码器		微程序控制器	
微程序入口查找逻辑		硬布线状态机	
条件判别测试逻辑		硬布线控制器	

表 1-4. 支持中断响应的单总线 CPU（现代时序）功能部件

## 1.3 实验步骤

- (1) MIPS 指令译码器：将 32 位输入操作码用分线器接出，最高 6 位为操作码 op，21-25 位为 rs 寄存器编号，16-20 位为 rt 寄存器编号、11-15 位为 rd 寄存器编号，0-5 位为功能码 func；取最高 6 位 op，跟需要实现的操作码常量

# 华中科技大学课程实验报告

进行比较，输出对应的指令信号。3 级时序单总线 CPU、现代时序单总线 CPU 和支持中断响应的现代时序单总线 CPU 具有相同的指令译码器，这里不再一一列出，统一给出。图 1-1 所示：

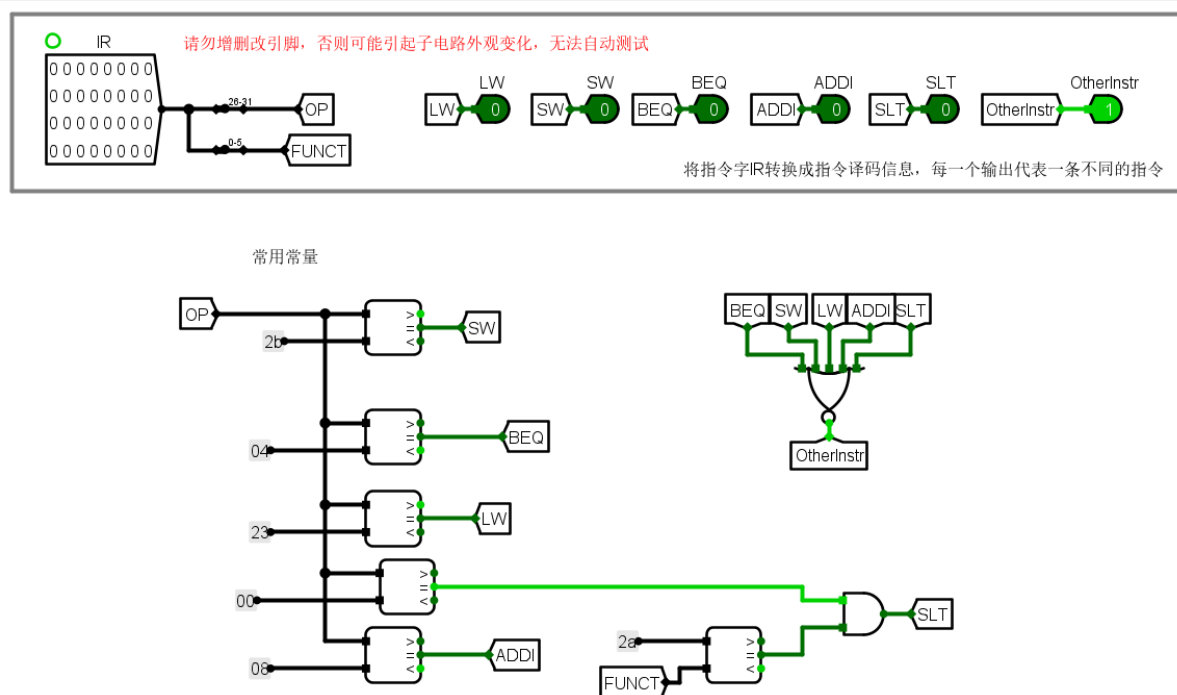


图 1-1 指令译码器

- (2) 时序发生器状态机：单总线结构中如果采用定长指令周期，所有 MIPS 指令都需要 3 个机器周期，每个机器周期 4 个时钟节拍，一共需要 12 个状态，状态图如图 1-2：

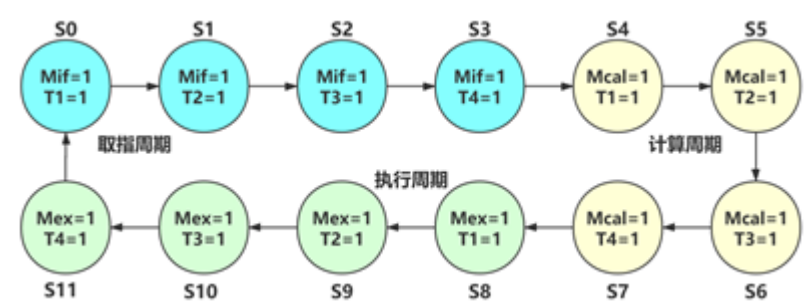


图 1-2 定长指令周期状态图

按状态图填写相应的 excel 表（图 1-3），自动生成次态逻辑表达式后，即可在 logisim 中自动生成该电路。

1	当前状态(现态)						输入信号							下一状态 (次态)					
2	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		
3	0	0	0	0	0								1	0	0	0	1		
4	0	0	0	1	1								2	0	0	1	0		
5	0	0	1	0	2								3	0	0	1	1		
6	0	0	1	1	3								4	0	1	0	0		
7	0	1	0	0	4								5	0	1	0	1		
8	0	1	0	1	5								6	0	1	1	0		
9	0	1	1	0	6								7	0	1	1	1		
10	0	1	1	1	7								8	1	0	0	0		
11	1	0	0	0	8								9	1	0	0	1		
12	1	0	0	1	9								10	1	0	1	0		
13	1	0	1	0	10								11	1	0	1	1		
14	1	0	1	1	11								0	0	0	0	0		

图 1-3 3 级时序状态转换表

- (3) 时序发生器输出函数：输出函数为组合逻辑，输入为状态寄存器的现态输出，输出为状态周期电位和节拍电位信号，定长指令周期的状态周期电位和节拍电位信号时序如图 1-4：

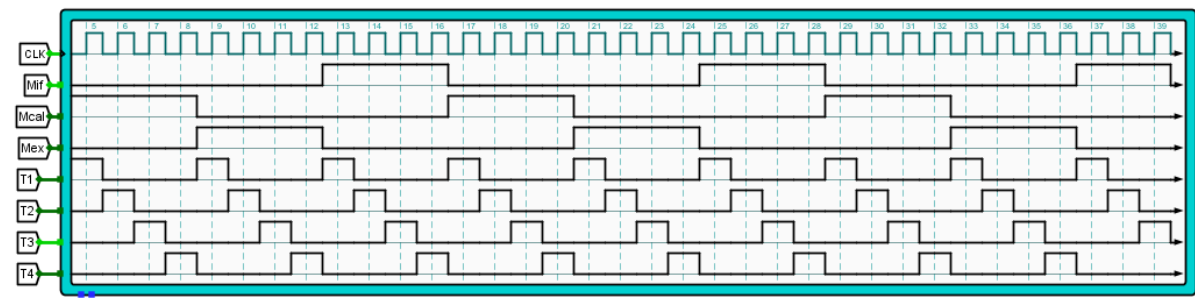


图 1-4 状态周期电位和节拍电位信号

根据时序输出要求，填写 excel 表（图 1-5），自动生成各输出信号的逻辑表达式，在 logisim 中自动生成电路即可。

1	当前状态(现态)					输出												
2	S3	S2	S1	S0	现态 10进制	Mif	Mcal	Mex	Mint	T1	T2	T3	T4	Out9	Out10	Out11	Out12	
3	0	0	0	0	0	1				1								
4	0	0	0	1	1	1					1							
5	0	0	1	0	2	1						1						
6	0	0	1	1	3	1							1					
7	0	1	0	0	4		1			1								
8	0	1	0	1	5		1				1							
9	0	1	1	0	6		1					1						
10	0	1	1	1	7		1						1					
11	1	0	0	0	8			1		1								
12	1	0	0	1	9			1			1							
13	1	0	1	0	10			1				1						
14	1	0	1	1	11			1					1					

图 1-5 输出函数真值表

- (4) 硬布线控制器组合逻辑单元：硬布线控制器组合逻辑单元生成的所有微操作控制信号都是反馈信号，指令译码信号，状态周期电位，节拍电位的组合



逻辑函数，如下式：

C\_n = \sum\_{m,i,k,j}(I\_m \cdot M\_i \cdot T\_k \cdot B\_j)

可以列出所有微操作信号的产生条件，填写下面的 excel 表格（图 1-6），自动生成逻辑表达式，然后再 Logisim 中自动生成电路。

1	输入（填1或0，不填为无关项x）														输出（只填写为1的情况）																						
2	Mif	Mcal	Mex	Mint	T1	T2	T3	T4	LW	SW	BEQ	SLT	ADDI	EQVAL	PCOut	DRout	Zout	Rout	WInet	WJInet	DREout	PCIn	ARIn	DREIn	DRIn	XIn	RIn	IRIn	PSWIn	Rs/Rt	RegDst	Add	Add4	Slt	READ	WRITE	
3	1				1										1								1				1										
4	1					1																1			1								1				1
5	1						1										1																				1
6	1							1									1												1								
7		1			1				1									1									1										
8		1				1				1									1														1				
9			1		1					1								1					1														
10			1			1																		1													1
11			1				1				1								1									1									
12				1							1								1								1										
13		1				1					1									1													1				
14			1		1						1							1					1														
15			1			1					1								1						1												
16			1				1				1										1																1
17		1			1						1								1								1										
18		1				1					1							1																1			
19			1		1						1					1											1										
20			1			1					1									1													1				
21			1				1				1							1					1														
22			1		1								1					1									1										
23			1			1							1							1														1			
24			1				1							1				1																			
25			1		1							1							1								1								1		
26			1			1							1						1																		
27			1				1					1						1										1									

图 1-6 组合逻辑真值表

- (5) 硬布线控制器（3 级时序）：连接设计好的时序发生器和硬布线控制器组合逻辑单元，实现输入指令码，输出微操作控制信号的逻辑功能。其中时序发生器的框架如图 1-7。

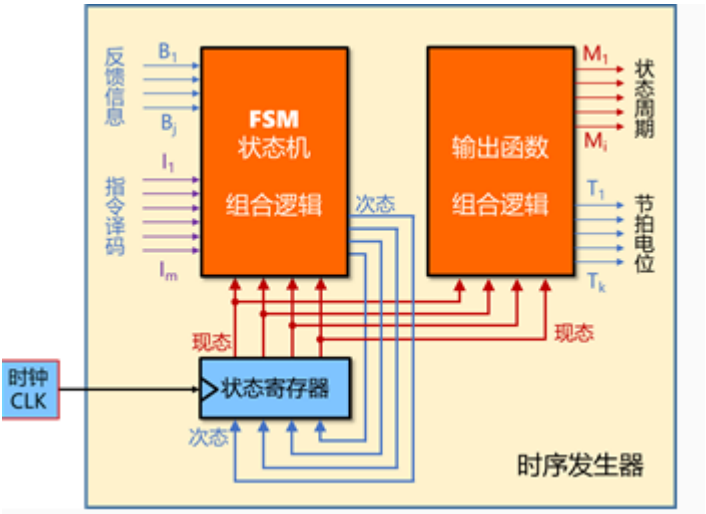


图 1-7 3 级时序发生器框架

# 华中科技大学课程实验报告

连接好的硬布线控制器（3级时序）如图 1-8 所示：

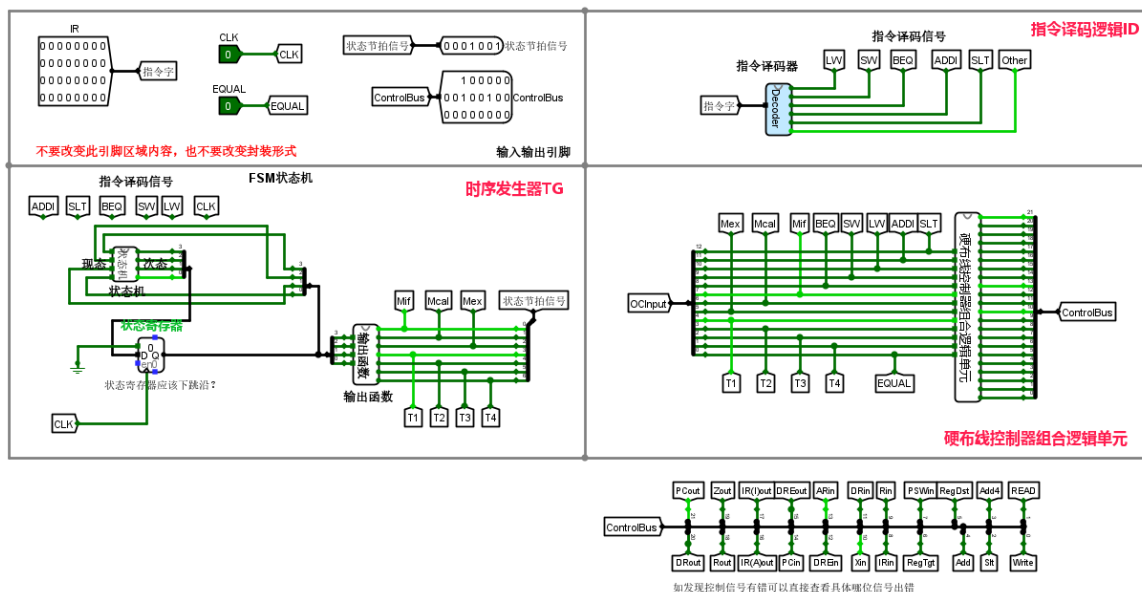


图 1-8 硬布线控制器（3级时序）

- (6) 微程序入口查找逻辑：在指令状态中，lw, sw, beq, add, addi 这五条指令对应的开始周期是 S4、S9、S14、S19、S22，所以入口地址分别是 4、9、14、19、22，在实现中断响应的 CPU 中还额外多了一条 ERET 指令，其地址入口是 25. 将其填入微程序入口地址表格, 获得逻辑表达式，利用 logisim 中的分析组合逻辑电路功能自动生成电路。分别如图 1-9 、图 1-10 所示：

机器指令译码信号					微程序入口地址					
LW	SW	BEQ	SLT	ADDI	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

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	
					1	25	1	1	0	0	1	

图 1-9 微程序入口地址（现代时序） 图 1-10 微程序地址入口（中断）

- (7) 条件判别测试逻辑：在现代时序中只有三位判别为 P0、P1、Equal, P0 为判别测试位，为 1 表示要根据指令功能进行微程序分支, P1 为 1 表示要根据 equal 标志进行微程序分支, Equal 为 1 表示运算相等；而在实现中断响应中多了 P2 和 IntR 两个判断位，P2 表示是最后一条微指令，需要判断中断请求 IntR, IntR 表示中断请求信号。两种组合逻辑的真值表分别如图 1-11、1-12 所示：

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
1	0				0	0	1
0	1		1			1	0
1	1		1			1	1

图 1-11 条件判别测试逻辑（现代时序）

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
1	0	0	0	1	0	0	1
1	0	0	1	0	0	0	1
1	0	0	1	1	0	0	1
0	1	0	1	0	0	1	0
0	1	1	1	0	0	1	0
0	1	1	1	1	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 1-12 条件判别测试逻辑（中断实现）

- (8) 微程序控制器：将微程序入口查找逻辑，判别测试逻辑和控制存储器等部件进行适当的连接，实现微程序控制器的主要数据通路，设计微程序并加载到控制存储器中。控制存储器中的数据由其控制信号和下地址字段构成。中断实现 CPU 中只是多了 ERET 和中断控制信号 IntR。如图 1-13、1-14 所示：

# 华中科技大学课程实验报告

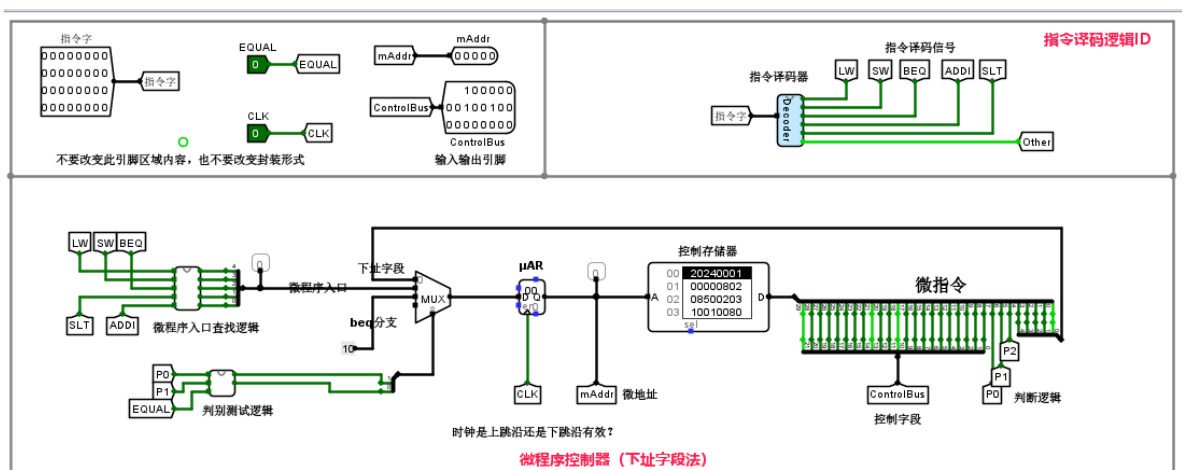


图 1-13 微程序控制器（现代时序）

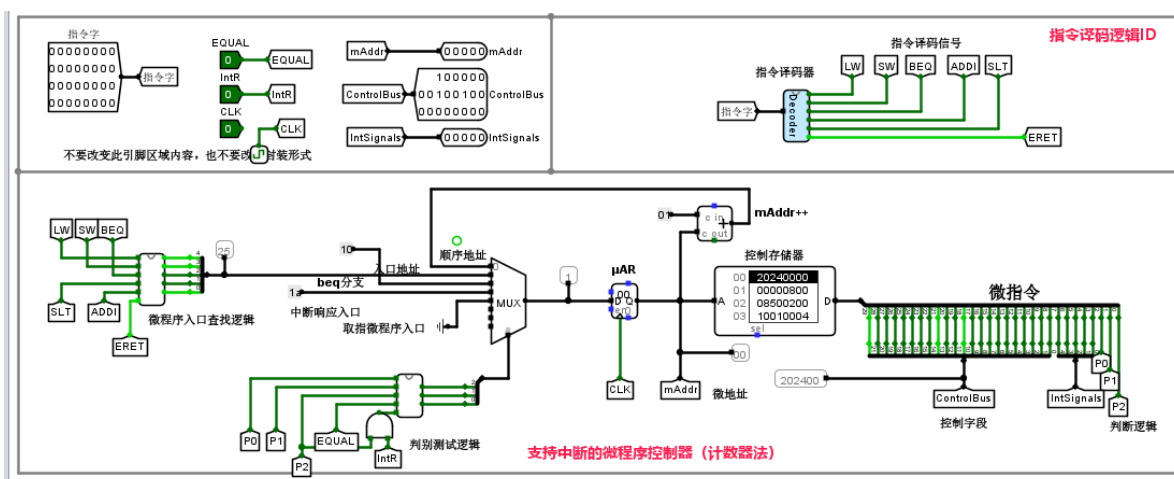
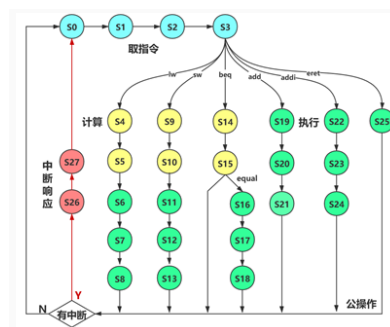
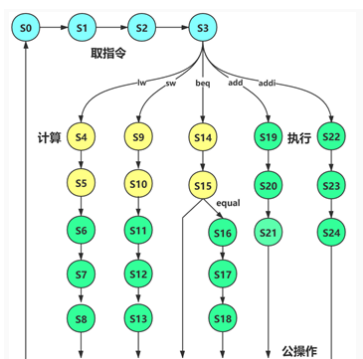


图 1-14 支持中断的微程序控制器

- (9) 硬布线状态机：根据现态和指令字，决定输出的次态。分别根据现代时序硬布线控制器状态图（图 1-15）和现代时序系统中硬布线控制器中断机制状态图（图 1-16）填写对应的 excel 表，自动生成次态逻辑表达式后，即可在 logisim 中自动生成电路。



# 华中科技大学课程实验报告

(10) **硬布线控制器**：根据 CPU 的现态输出对应的微程序控制信号。与微程序控制器不同的是，硬布线控制器更加的简洁方便，但是对于指令字的可扩展性差。硬布线控制器和支持中断机制的硬布线控制器分别如图 1-17、图 1-18 所示：

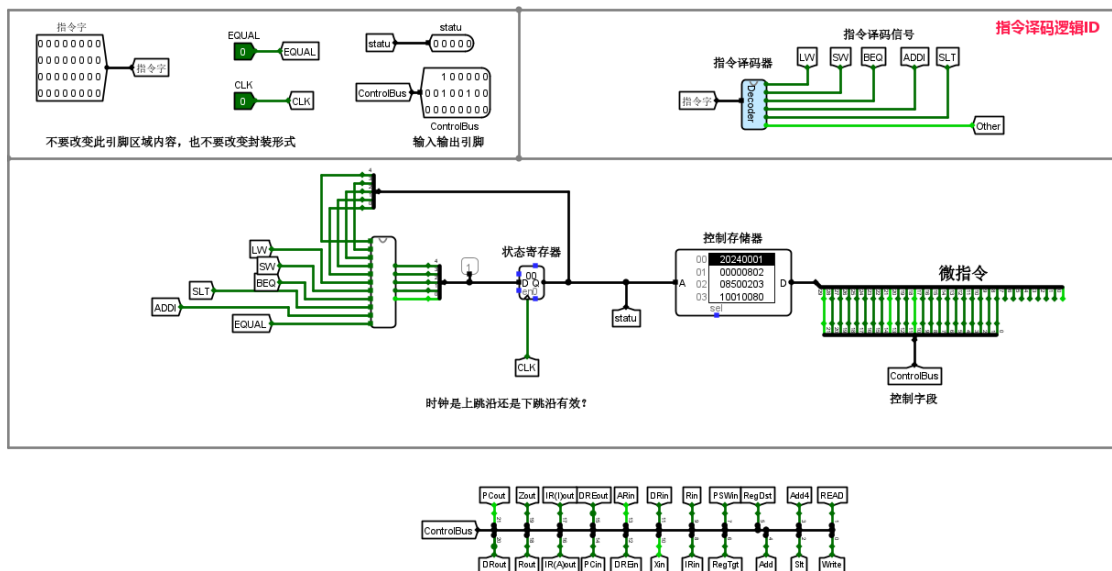


图 1-17 硬布线控制器（现代时序）

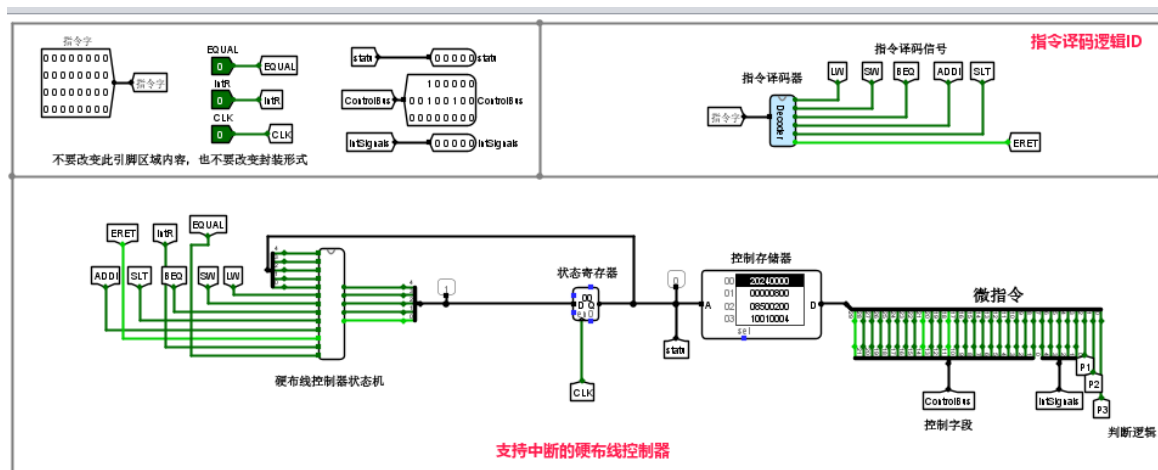


图 1-18 支持中断的硬布线控制器

(11) 单总线 CPU (定长指令周期 3 级时序): 在完成所有的器件设计之后, 将对应的数据通路和控制器, 寄存器相连, 就可得到 CPU 全貌, 如图 1-19 所示:



# 华中科技大学课程实验报告

(13) 支持中断机制单总线 CPU（现代时序）：支持中断机制的 CPU 比普通 CPU 多了中断控制器和中断使能信号寄存器以及 EPC，用于保护现场和恢复现场，CPU 全貌如图 1-21 所示：

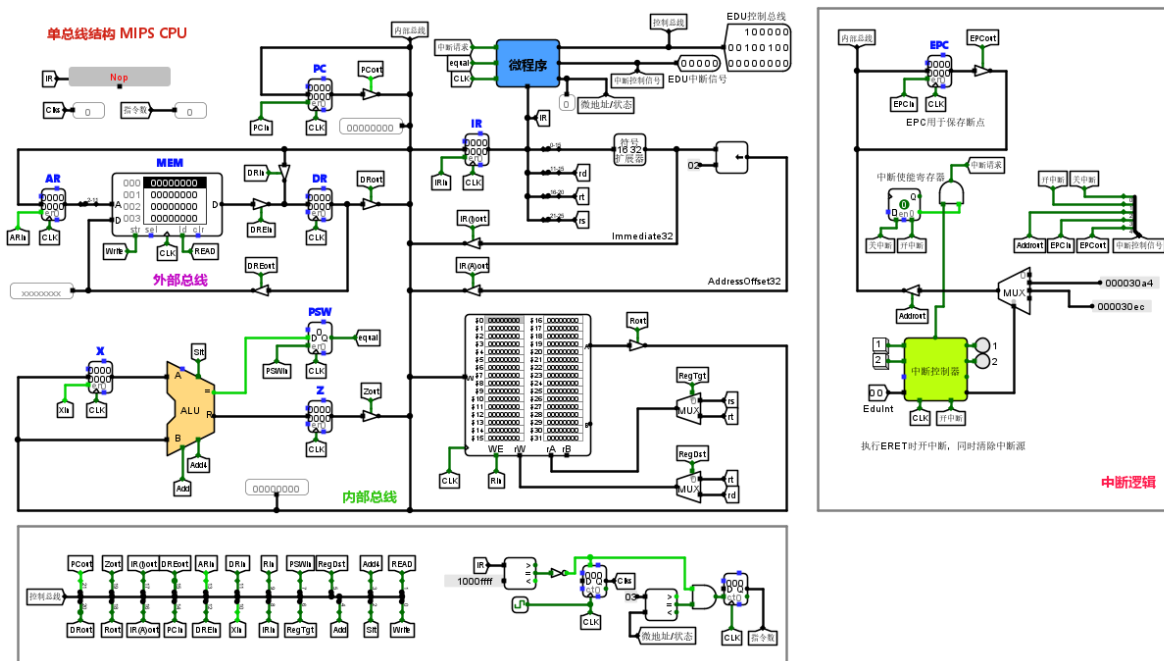


图 1-21 支持中断机制现代时序单总线 CPU

## 1.4 故障与调试

### 1.4.1 微指令生成错误

**故障现象:** 填写单总线现代时序微程序控制设计表显示错误

微指令名称	PCout	DRout	Zout	float	float	float	float	float	PCin	ARin	DRin	Xin	Rin	IRin	PSW	Op	Op2	Op3	Add	Addr	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	Rin <td>IRin</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>1</td> <td>10000000100100000000000000000001</td> <td>20240001</td>	IRin												1	10000000100100000000000000000001	20240001	
取指令	1																				1					2	00000000000000000000000000000010	802	
取指令	2		1							1	1											1				3	00100001010000000000000000000011	8500203	
取指令	3	1												1										1		0	01000000000000000000000000000000	10010080	
LOAD指令	4				1							1													5	00010000000100000000000000000101	4040005		
LOAD指令	5				1																1				6	00001000000000000000000000000110	2001006		
LOAD指令	6			1						1																7	00100000100000000000000000000111	8200007	
LOAD指令	7									1												1				8	00000000100000000000000000000100	100208	
MOVE指令	8	1											1													0	01000000000000000000000000000000	10020000	
ADD指令	9				1							1														10	000100000001000000000000000001010	404000A	
ADD指令	10				1																1					11	000010000000000000000000000001011	2001008	
ADD指令	11			1						1																12	001000001000000000000000000001100	820000C	
STORE指令	12			1							1							1								13	000100000010000100000000000001101	4084000	
STORE指令	13						1					1											1			0	00000010000000000000000000000000	800100	
STORE指令	14				1							1														15	000100000001000000000000000001111	404000F	
JMP指令	15				1											1	1					1				16	0001000000000001100010001010000	400450	
	16	1										1														17	10000000000100000000000000000001	20040011	
	17						1													1						18	00000100000000000000000000000010	1001012	
	18			1						1																0	00100001000000000000000000000000	8400000	
	19				1							1														20	00010000000100000000000000000100	4040014	
	20				1								1					1					1			21	000100000000000100010000010101	4004415	
	21			1									1						1							0	00100000000100010000000000000000	8022000	
	22				1							1														23	00010000000100000000000000000111	4000017	
	23				1								1									1				24	00001000000000000000000000000100	2001018	
	24			1								1														0	00100000000010000000000000000000	8020000	
																												错误	

图 1-22 单总线现代时序微程序控制器设计图



# 华中科技大学课程实验报告

**原因分析：**如图 1-22，在状态地址为 15 号的微程序中，执行 `Jmp` 指令时，下地址时 0 而不是 16，且在 `Jmp` 时，并不需要 `slt` 的控制信号。

**解决方案：**去掉 `slt` 控制信号，且将下地址修改为 0。

## 1.5 测试与分析

### 1.5.1 单总线 CPU（3 级时序）执行 `sort-5.hex` 文件

(1) 内存布局，在 80 号单元开始处出现 6,5,4,3,2,1,0,ffff 的有符号降序数据，如图 1-23 所示：

```
000 2010ffff 20110000 ae300200 22100001 22310004 ae300200 22100001 22310004
008 ae300200 22100001 22310004 ae300200 22100001 22310004 ae300200 22100001
010 22310004 ae300200 22100001 22310004 ae300200 22100001 22310004 ae300200
018 20100000 2011001c 8e130200 8e340200 0274402a 11000002 ae330200 ae140200
020 2231ffff 12110001 1000ffff 22100004 2011001c 12110001 1000ffff 1000ffff
028 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
030 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
038 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
040 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
048 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
050 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
058 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
060 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
068 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
070 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
078 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
080 00000006 00000005 00000004 00000003 00000002 00000001 00000000 ffffffff
```

图 1-23 单总线 CPU（3 级时序）执行 `sort-5.hex` 后内存布局

(2) 时钟周期，执行完毕后，最后一条指令是一条 `beq` 分支指令，会跳回当前指令继续执行，是死循环，`sort-5.hex` 的时钟周期数是 251，如图 1-24 所示：

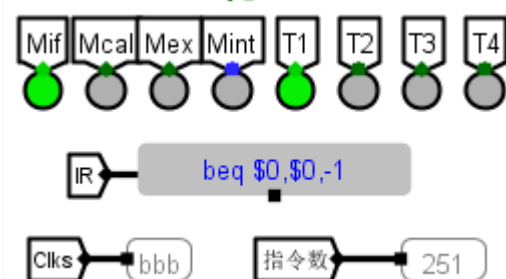


图 1-24 单总线 CPU（3 级时序）执行 `sort-5.hex` 所需时钟周期数

### 1.5.2 单总线 CPU（现代时序）执行 `sort-5.hex` 文件

(1) 内存布局，在 80 号单元开始处出现 6,5,4,3,2,1,0,ffff 的有符号降序数据，如



图 1-25 所示:

```

000 2010ffff 20110000 ae300200 22100001 22310004 ae300200 22100001 22310004
008 ae300200 22100001 22310004 ae300200 22100001 22310004 ae300200 22100001
010 22310004 ae300200 22100001 22310004 ae300200 22100001 22310004 ae300200
018 20100000 2011001c 8e130200 8e340200 0274402a 11000002 ae330200 ae140200
020 2231fffc 12110001 1000fff7 22100004 2011001c 12110001 1000fff3 1000ffff
028 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
030 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
038 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
040 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
048 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
050 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
058 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
060 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
068 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
070 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
078 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
080 00000006 00000005 00000004 00000003 00000002 00000001 00000000 ffffffff
    
```

图 1-25 单总线 CPU（现代时序）执行 sort-5.hex 后内存布局

(2) 时钟周期，执行完毕后，最后一条指令是一条 beq 分支指令，会跳回当前指令继续执行，是死循环，sort-5.hex 的时钟周期数是 251，如图 1-26 所示:



图 1-26 单总线 CPU（现代时序）执行 sort-5.hex 所需时钟周期数

### 1.5.3 支持中断机制的单总线 CPU（现代时序）执行 sort-5.hex 文件

(1) 内存布局，在 80 号单元开始处出现 6,5,4,3,2,1,0,ffff 的有符号降序数据，如图 1-27 所示:

```

000 2010ffff 20110000 ae300200 22100001 22310004 ae300200 22100001 22310004
008 ae300200 22100001 22310004 ae300200 22100001 22310004 ae300200 22100001
010 22310004 ae300200 22100001 22310004 ae300200 22100001 22310004 ae300200
018 20100000 2011001c 8e130200 8e340200 0274402a 11000002 ae330200 ae140200
020 2231fffc 12110001 1000fff7 22100004 2011001c 12110001 1000fff3 1000ffff
028 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
030 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
038 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
040 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
048 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
050 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
058 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
060 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
068 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
070 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
078 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
080 00000006 00000005 00000004 00000003 00000002 00000001 00000000 ffffffff
    
```

# 华中科技大学课程实验报告

图 1-27 单总线 CPU（现代时序）执行 sort-5.hex 后内存布局

(2) 时钟周期，执行完毕后，最后一条指令是一条 beq 分支指令，会跳回当前指令继续执行，是死循环，sort-5.hex 的时钟周期数是 251，如图 1-28 所示：



图 1-28 单总线 CPU（现代时序）执行 sort-5.hex 所需时钟周期数

(3) 支持中断处理机制：当我按下 1 的开关申请中断响应时，中断申请 1 所对应的信号灯会变红，如图 1-29、图 1-30 所示：

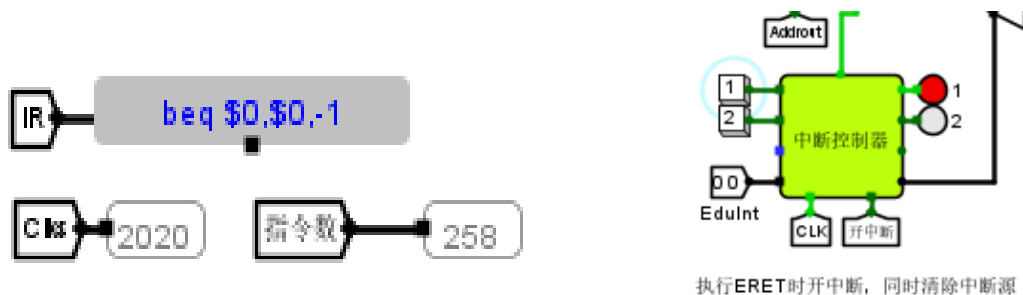


图 1-29 中断申请是指令数

图 1-30 中断申请 1 的信号灯变红

当完成中断申请是指令数会加 1，且信号灯恢复，如图 1-31 所示：

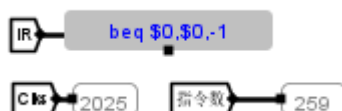


图 1-31 执行完中断响应后指令数

## 2 总结与心得

### 2.1 实验总结

本次实验主要完成了如下几点工作：

- 1) 实现了 32 位定长指令的译码
- 2) 实现了对各个寄存器，存储器的读写
- 3) 实现了控制信号的生成
- 4) 使用了多路选择器实现了对多个数据来源的选择和控制
- 5) 实现了单总线 CPU（3 级时序）的硬布线控制器
- 6) 实现了单总线 CPU（现代时序）的微程序控制器和硬布线控制器
- 7) 实现了支持中断机制的单总线 CPU（现代时序）的微程序控制器和硬布线控制器
- 8) 完善了单总线 CPU（3 级时序）、单总线 CPU（现代时序）和支持中断机制的单总线 CPU（现代时序）的综合数据通路，实现了简单的冒泡排序的逻辑功能。

### 2.2 实验心得

- 1) 熟悉了使用 logisim 的一些基础功能和一些 较高阶的功能。
- 2) 熟悉的掌握了 CPU 的各个模块的功能和各个模块的逻辑实现以及具体的电路设计。
- 3) 在实验中对于课堂上面讲述的关于 CPU 的知识有了更加深刻的理解，在课堂上面对于这方面的知识永远是停留在纸面的，只有自己真正的动手实践了，才知道具体的功能是如何实现的，才知道 CPU 内部各个运算器件如何协同工作来完成 CPU 的整个功能。
- 4) 在实验的过程中遇到了不少的问题，不过在组原教学交流群里面有老师在我们的实验的过程中对于我们提出的问题都给出了很及时和详细的解答，同时群里面也有很多同学们在热心的回答我们的问题，所以我们提出的问题可能有些同学也刚好遇到过，因此我们的难题能够及时地得到解答。

# 华中科技大学课程实验报告

---

- 5) 最后总结下来，在计算机组成原理的实验课程中确实掌握了一些硬知识，结合自己以前所学习的硬件语言和这学期的操作系统，还是对整个计算机专业方面的相关知识有了更深的了解。

## 参考文献

- [1] DAVID A. PATTERSON(美). 计算机组成与设计硬件/软件接口(原书第 5 版). 北京:机械工业出版社.
- [2] David Money Harris(美). 数字设计和计算机体系结构(第二版). 机械工业出版社
- [3] 谭志虎, 秦磊华, 胡迪青. 计算机组成原理实践教程. 北京:清华大学出版社, 2018 年.
- [4] 秦磊华, 吴非, 莫正坤. 计算机组成原理. 北京:清华大学出版社, 2011 年.
- [5] 袁春风编著. 计算机组成与系统结构. 北京:清华大学出版社, 2011 年.
- [6] 张晨曦, 王志英. 计算机系统结构. 高等教育出版社, 2008 年.

• 指导教师评定意见 •

---

### 一、原创性声明

本人郑重声明本报告内容，是由作者本人独立完成的。有关观点、方法、数据和文献等的引用已在文中指出。除文中已注明引用的内容外，本报告不包含任何其他个人或集体已经公开发表的作品成果，不存在剽窃、抄袭行为。

特此声明！

作者签字: 嵌入签名图片

### 二、对课程实验的学术评语（教师填写）

### 三、对课程实验的评分（教师填写）

评分项目 (分值)	报告撰写 (30 分)	课设过程 (70 分)	最终评定 (100 分)
得分			

指导教师签字: \_\_\_\_\_