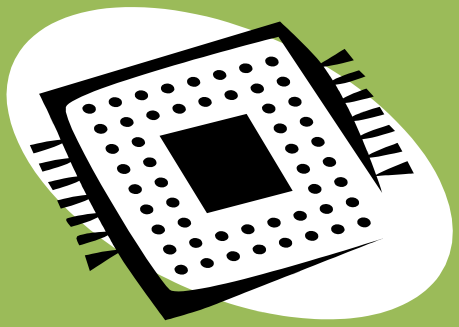
华中科技大学

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：	U201814670
姓　　名：	李田田
电　　话：	15623203778
邮　　件：	1240815169@qq.com
完成日期：	2020-12-11



计算机科学与技术学院

目 录

1 单总线 CPU（3 级时序）设计实验

1.1 设计要求

1. MIPS 指令译码器设计：利用比较器等功能模块将 32 位 MIPS 指令字译码生成 LW、SW、BEQ、SLT、ADDI、OtherInstr 信号。

2. 定长指令周期---时序发生器 FSM 设计：利用数字逻辑电路相关知识设计定长指令周期的三级时序系统。

3. 定长指令周期---时序发生器输出函数设计：利用数字逻辑电路相关知识设计定长指令周期的三级时序系统，时序发生器包括状态机和输出函数两部分。

4. 硬布线控制器组合逻辑单元：在实现了指令译码逻辑、时序发生器主要功能部件后，进一步设计实现控制器核心模块硬布线控制器组合逻辑单元，控制器框架如下图所示。

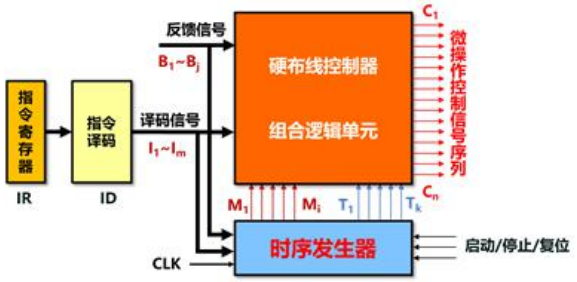


图 1.1 硬布线控制器

5. 定长指令周期---硬布线控制器设计：实现硬布线控制器的集成，其中时序发生器框架如下图：

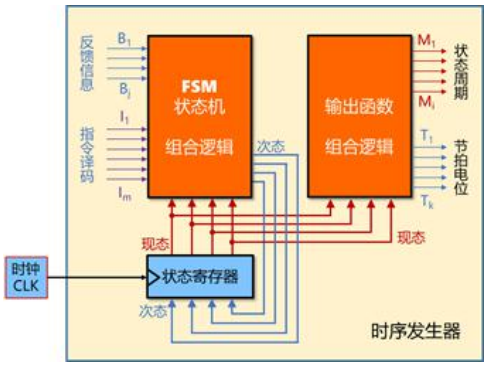


图 1.2 时序发生器

6. 定长指令周期---单总线 CPU 设计：在 RAM 中加载 sort-5.hex 程序，ctrl+k 自动运行，程序应该运行至 0xbbb 节拍停下，指令计数为 251，注意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。

1.2 方案设计

1.2.1 指令译码器

通过将 op 操作位与这几个指令的操作位比较，如果相同，则为该指令，否则为其他指令。其中 SLT 指令需要将 op 与操作位比较完之后再与 FUNCT 进行比较，才能得到是否为该指令。

1.2.2 时序发生器状态机

设计状态机时，通过实验所给状态迁移图，如图 1.3 所示。通过该图填写所给的 excel 表，填写结果如图 1.4 所示，将真值表所得的表达式填写到分析电路时对应的位置，即可生成对应的电路图。

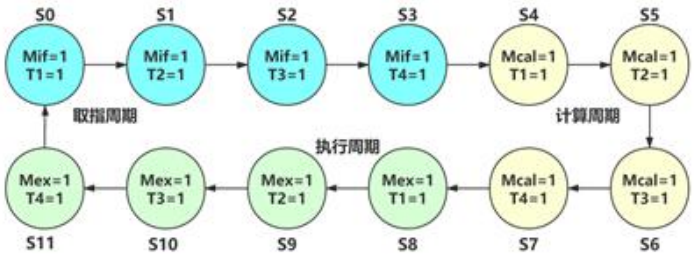


图 1.3 状态迁移图

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								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	1	1	0	6								7	0	1	1	1
0	1	1	1	7								8	1	0	0	0
1	0	0	0	8								9	1	0	0	1
1	0	0	1	9								10	1	0	1	0
1	0	1	0	10								11	1	0	1	1
1	0	1	1	11								0	0	0	0	0

图 1.4 状态机真值表

1.2.3 时序发生器输出函数

通过图 1.3 所给的每个状态，填写对应的 excel 表格，得到如图 1.5 所示的表格，将表格生成的表达式填写到对应的分析电路的对应位置，生成电路即可得到时序发生器输出函数的电路图。

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			
1	0	0	1	9			1			1		
1	0	1	0	10			1				1	
1	0	1	1	11			1					1

图 1.5 输出函数真值表

1.2.4 硬布线控制器组合逻辑单元

硬布线控制器的组合逻辑单元控制在在每个状态，如果收到什么样的信号，输出对应位的指令。其中前四个取指令操作是所有指令共享的，都相同。但是对于后面周期的指令，需要每个指令分开写，因为它们输出的指令不同。最终得到的表格如图 1.6 所示，其中注意 SLT 指令其前部分操作与 addi 一致，但在 Mint T2 周期时，其输出指令有一个 Slt 的操作，这点与 PPT 并不一致，注意修改。

输入 (填1或0，不填为无关项x)												输出 (只填写为1的情况)																							
Mif	Mcal	Mex	Mint	T1	T2	T3	T4	LW	SW	BEQ	SLT	ADDI	READ	PCout	DRout	Zout	Rout	MDRout	IRAOut	OffSet	PCin	ARin	DRBin	DRin	Xin	Rin	IRin	PSWin	Rx/REi	RegDist	Add	Add4	Slt	READ	WRITE	ControlBus (hex)
1				1										1								1			1											202400
1					1																											1				8
1						1										1					1		1										1			85002
1							1								1											1										100100
	1			1				1									1								1											40400
	1				1			1										1													1					20010
	1			1					1								1								1											40400
	1				1				1									1													1					20010
	1			1						1							1								1											40400
	1				1					1							1													1	1					400C0
		1		1				1								1						1														82000
		1			1			1															1											1			1002
		1				1		1								1										1										100200
		1		1					1							1						1														82000
		1			1				1								1						1						1							40840
		1				1			1											1														1		8001
		1		1						1				1											1											200400
		1			1					1							1														1					10010
		1				1				1			1			1					1																84000
		1		1								1					1								1											40400
		1			1							1						1													1					20010
		1				1						1				1										1										80200
		1		1							1						1								1											40400
		1			1						1						1												1				1			40044
		1				1					1					1										1				1						80220

图 1.6 组合逻辑单元真值表

1.2.5 硬布线控制器

由图 1.1 的硬布线控制器的逻辑图可知，状态寄存器将状态传给状态机和输出函数，状态机将次态传给状态寄存器作为输入，使用分线器连接各个点得到如图 1.7 所示。

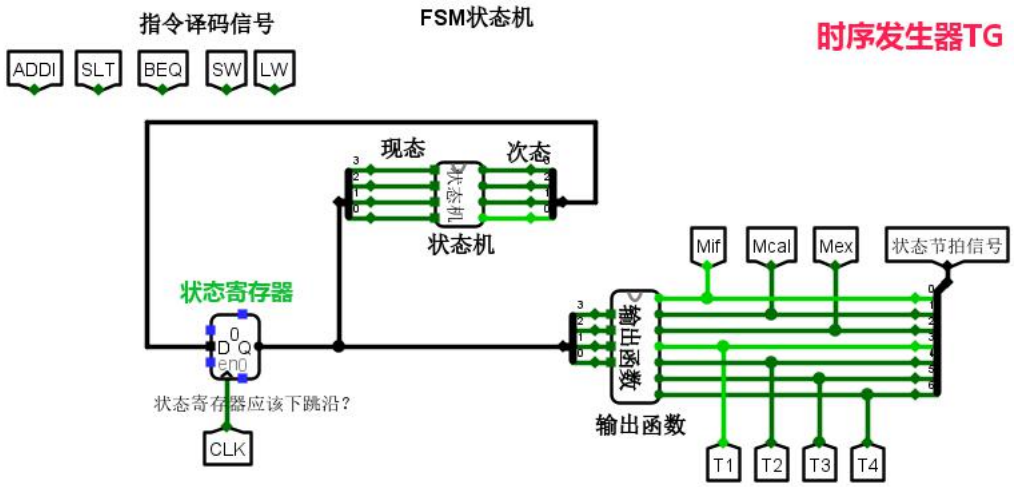


图 1.7 硬布线控制器

1.3 实验步骤

1） 按照题目要求对于整个系统的架构进行分析，将 cpu 分成多个部件然后组装到一起，根据每一关构造各个模块，按照平台实验要求和老师对应的视频文件，利用对应的 excel 表完成对应的 circ 电路文件，并在每个模块构建完成后对其进行单元测试，以保证各个模块独自工作时功能是正常的。

2） 将完成的 circ 电路图用记事本打开，将代码复制后放入平台测试每一关。

3） 对应平台中测试机不同的测试集报错信息对自己的电路图进行修改。 在构建好的 CPU 上加载程序并运行，对于执行错误的地方进行单步调试并分析错误原因，然后对于电路进行修改，直到没有任何错误出现。

1.4 故障与调试

在进行单总线 CPU 的测试时，打开 MEM 的存储内容，发现排序结果与老师给的

并不一致，经资料和老师讲解后明白因为 Z 寄存器没有控制，定长周期的空周期导致 Z 值错误，得到的排序跑到了最开头的位置。

1.5 测试与分析

加载测试镜像 sort-5.hex，得到的结果如图 1.8 所示，指令数停在 251。得到的排序结果如图 1.9 所示，与要求实验结果不一致，分析如下：

在进行单总线 CPU 的测试时，打开 MEM 的存储内容，发现排序结果与老师给的并不一致，经资料和老师讲解后明白因为 Z 寄存器没有控制，定长周期的空周期导致 Z 值错误，得到的排序跑到了最开头的位置。

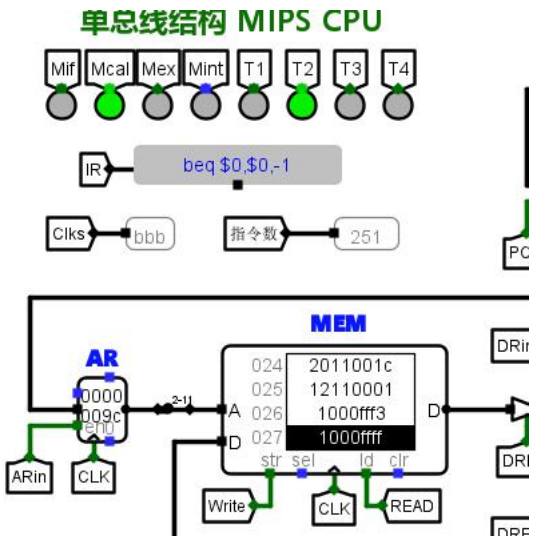


图 1.8 单总线 CPU

000	00000006	00000005	00000004	00000003	00000002	00000001	00000000	ffffffff	ae300200	22100001	22310004	ae300200	22100001	22310004	ae300200	22100001
010	22310004	ae300200	22100001	22310004	ae300200	22100001	22310004	ae300200	20100000	2011001c	8e130200	8e340200	0274402a	11000002	ae330200	ae140200
020	2231fffc	12110001	1000fff7	22100004	2011001c	12110001	1000fff3	1000ffff	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
030	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
040	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
050	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
060	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
070	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
080	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000

图 1.9 测试程序排序结果

2 单总线 CPU（现代时序）设计实验

2.1 设计要求

1. 单总线 CPU 微程序入口查找逻辑：设计如下电路，根据指令译码信号生成 5 位的微程序入口地址。

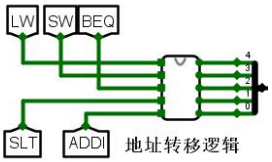


图 2.1 微程序地址转移逻辑

2. 单总线 CPU 微程序条件判别测试逻辑：根据微指令字中的判别测试字段和条件反馈信息生成后续地址的多路选择信号，要求实现对应组合逻辑。

3. 单总线 CPU 微程序控制器设计：将微程序入口查找逻辑,判别测试逻辑，控制存储器等部件进行适当连接设计微程序并加载到控制存储器中。

4. 采用微程序的单总线 CPU 设计：在 RAM 中加载 sort-5.hex 程序，ctrl+k 自动运行，程序应该运行至 0x7c1 节拍停下，指令计数为 251。

5. 现代时序硬布线控制器设计：在实现指令译码、现代时序状态机模块后，最终实现硬布线控制器的集成，完成硬布线控制器框架连接。

6. 现代时序硬布线控制器状态机设计：利用数字逻辑电路相关知识设计现代时序硬布线核心部件状态机模块,实际状态机如下图：

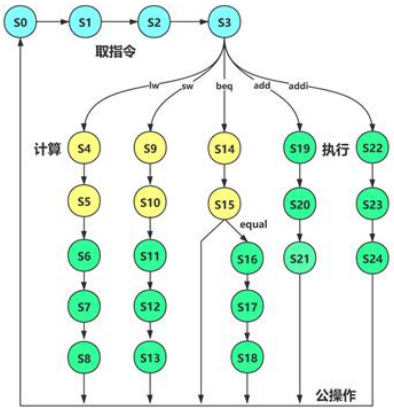


图 2.2 硬布线状态机

2.2 方案设计

2.2.1 单总线 CPU 微程序入口查找逻辑

根据图 2.2 的状态转移图，发现入口状态为 4、9、14、19、22。根据每个入口状态对应的指令，填写 ecxcel 表，得到如图 2.3 所示的真值表，将得到的表达式写入对应的分析电路中，得到该电路。

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

图 2.3 入口逻辑真值表

2.2.2 微程序条件判别测试逻辑

P0：输入判别测试位，为 1 表示要根据指令功能进行微程序分支。P1：输入判别测试位，为 1 表示要根据 equal 标志进行微程序分支。Equal：输入条件状态位，表示运算相等。在电路图中可以看出，00 为下址字段，01 为微程序入口，10 为 beq 分支，所以当 P0 为 1 时，下条指令从微程序入口进入，即 01；当 P1 为 1 时，要判别 equal 是否为 1，为 1 则下条指令从 10 进入，为 0 则下条指令从 00 进入。

2.2.3 微程序控制器

判别测试逻辑得到的两位为多路选择器的选择条件，下址字段为微指令的后五位，即下个状态，微程序入口为根据指令功能得到的下个状态，beq 分支为收到 equal 信号要执行的下个状态。电路图如图 2.4 所示。其中控制存储器的存储数据应该为微程序指令，该内容通过填写微程序指令 excel 表获得。如图 2.5 所示。

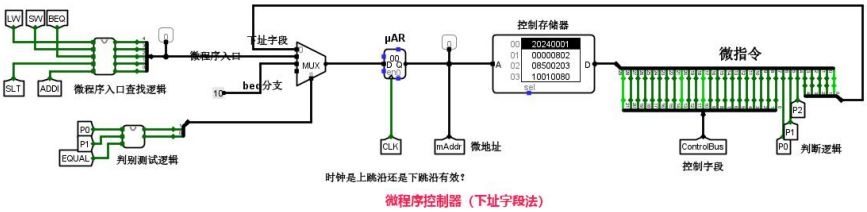
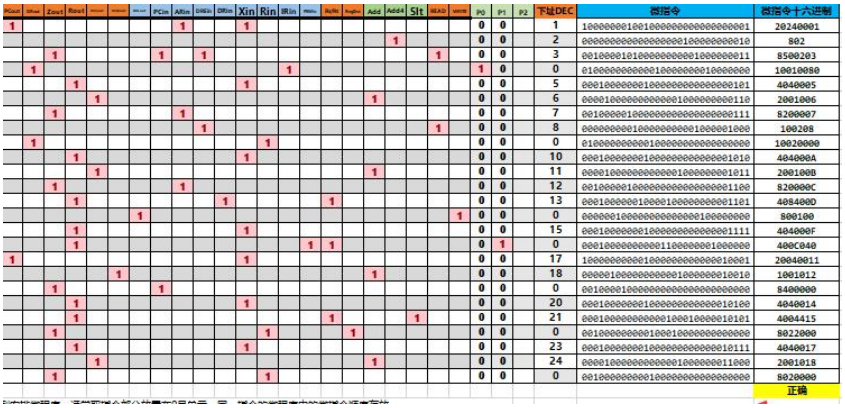


图 2.4 微程序控制器



Xin	Rin	P0	P1	P2	下址DEC	指令	指令十六进制
1		0	0		1	10000000100100000000000000000001	20240001
		0	0		2	00000000000000000001000000000010	802
	1	0	0		3	00100001010000000001000000000011	8500203
1		1	0		0	01000000000001000000010000000000	10010080
	1	0	0		5	00010000001000000000000000000101	4040005
		0	0		6	00001000000000001000000000000110	2001006
	1	0	0		7	00100000100000000000000000000111	8200007
		0	0		8	00000000010000000001000001000000	100208
1		0	0		0	01000000000010000000000000000000	10020000
	1	0	0		10	00010000001000000000000000001010	404000A
		0	0		11	00001000000000001000000000001011	200100B
	1	0	0		12	00100000100000000000000000001100	820000C
		0	0		13	00010000010000100000000000001101	408400D
		0	0		0	00000010000000000000001000000000	800100
		0	0		15	00010000001000000000000000001111	404000F
		0	1		0	00010000000000110000001000000000	400C040
1		0	0		17	10000000001000000000000000010001	20040011
		0	0		18	00000010000000001000000010010010	1001012
	1	0	0		0	00100001000000000000000000000000	8400000
		0	0		20	00010000001000000000000000010100	4040014
		0	0		21	00010000000000010001000000010101	4004415
	1	0	0		0	00100000000010001000000000000000	8022000
		0	0		23	00010000001000000000000000010111	4040017
		0	0		24	00001000000000001000000011000	2001018
	1	0	0		0	00100000000010000000000000000000	8020000
							正确

图 2.5 微程序指令表

2.2.4 现代时序硬布线控制器状态机

根据图 2.2 状态机的转移图，可以填写 excel 真值表，将填写完得到的公式放入分析电路的公式中，可得到状态机的电路图。

2.2.5 现代时序硬布线控制器

根据图 2.8 硬布线控制器的逻辑图，状态机将次态传给状态寄存器，状态寄存器将得到的现态传给状态机，得到如图 2.9 所示的电路图。

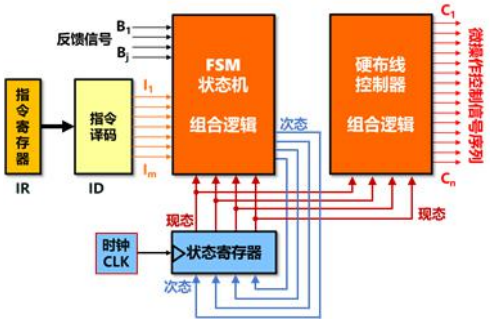


图 2.8 硬布线控制器逻辑

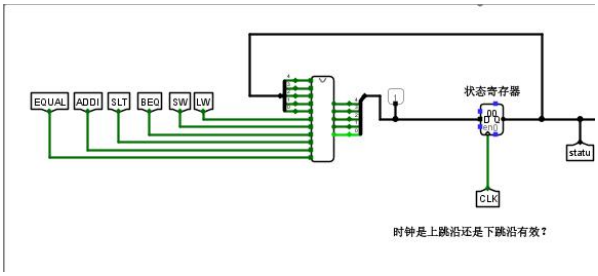


图 2.9 硬布线控制器电路图

2.3 实验步骤

4） 按照题目要求对于整个系统的架构进行分析，将 cpu 分成多个部件然后组装到一起，根据每一关构造各个模块，按照平台实验要求和老师对应的视频文件，利用对应的 excel 表完成对应的 circ 电路文件，并在每个模块构建完成后对其进行单元测试，以保证各个模块独自工作时功能是正常的。

5） 将完成的 circ 电路图用记事本打开，将代码复制后放入平台测试每一关。

6） 对应平台中测试机不同的测试集报错信息对自己的电路图进行修改。 在构建好的 CPU 上加载程序并运行，对于执行错误的地方进行单步调试并分析错误原因，然后对于电路进行修改，直到没有任何错误出现。

2.4 故障与调试

在设计微程序指令时，按照老师所给的表格发现怎么也不能通过测试，最后经过查找数据集调试发现，有些指令与老师给的并不一致，比如 SLT 指令其前部分操作与 addi 一致，但在 Mint T2 周期时，其输出指令有一个 Slt 的操作。

2.5 测试与分析

将设置好的微程序控制器放到 CPU 通路中进行测试，得到的结果如图 2.6 和图 2.7 所示，该结果正确，符合要求。

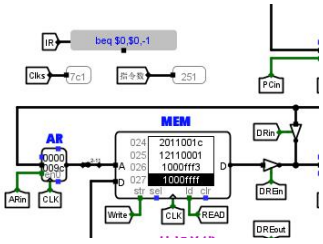


图 2.6 微程序单总线 CPU 结果

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

图 2.7 微程序单总线 CPU 结果

3 现代时序中断机制设计实验

3.1 设计要求

1. 支持中断微程序入口查找逻辑:根据指令译码信号生成 5 位的微程序入口地址。

2. 支持中断的微程序条件判别测试逻辑：根据微指令字中的判别测试字段和条件反馈信息生成后续地址的多路选择信号，要求实现对应组合逻辑。

3. 支持中断的微程序控制器设计：将微程序入口查找逻辑,判别测试逻辑，控制存储器等部件进行适当连接，设计微程序并加载到控制存储器中。

4. 支持中断的微程序单总线 CPU 设计：异常程序地址计数器 EPC，中断使能寄存器 IE，中断控制器等模块，将这些模块进行有效连接，并在本关进行最终的联调。

5. 支持中断的现代时序硬布线控制器状态机设计：利用数字逻辑电路相关知识设计现代时序硬布线核心部件状态机模块。

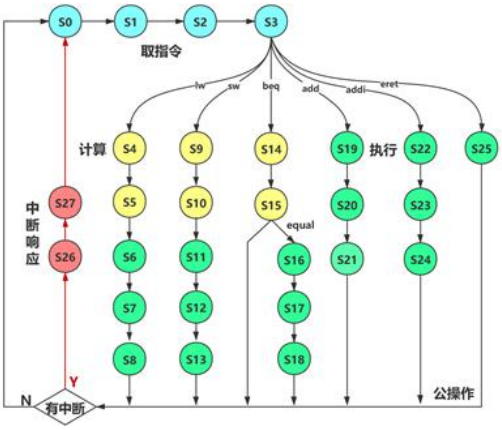


图 3.1 支持中断状态转移图

6. 支持中断的现代时序硬布线控制器设计：在实现指令译码、现代时序状态机模块后，最终实现硬布线控制器的集成，完成硬布线控制器框架连接。

3.2 方案设计

3.2.1 支持中断的微程序入口查找逻辑

对于支持中断的微程序入口查找逻辑，其对应的真值表为图 3.1 所示。因为要支

持中断，一共有 27 个状态需要选择，所以对应的入口地址为五位 S0~S4。每一个对应的指令信号都能进入不同的状态入口，其中与上面微程序不一样的是多了指令 ERET，这个指令控制了什么时候进入状态 25。

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 3.2 支持中断的微程序入口查找逻辑

3.2.2 支持中断的微程序条件判别逻辑

对于支持中断的微程序条件判别逻辑，在控制器中，多路选择器有五条输入路线可以选择，故 S 为三位，而该电路便是找出什么情况下选择哪条路径。

P0：输入判别测试位，为 1 表示要根据指令功能进行微程序分支；P1：输入判别测试位，为 1 表示要根据 equal 标志进行微程序分支；P2：输入判别测试位，为 1 表示是微程序的最后一条微指令，可能需要进行中断响应；Equal：输入条件状态位，表示运算相等。

当 P0 为 1 时，需要选择指令入口地址，s 为 001；P1 为 1 时，要判断 equal 是否为 1，为 1 则 S 为 010，P2 为 1 时，需要判断 INTR 是否为 1，若为 1，则进入中断相应入口 011。其中只要为最后一条指令 P2 就为 1。真值表如图 3.2 所示

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

图 3.3 支持中断的微程序条件判别逻辑

3.2.3 支持中断的微程序控制器设计

判别测试逻辑得到的两位为多路选择器的选择条件，顺序地址为下个状态，微程序入口为根据指令功能得到的下个状态，beq 分支为收到 equal 信号要执行的下个状态，中断响应入口为中断程序的地址，取址微程序入口为 0 即为程序取址操作的开始。电

路图如图 3.3 所示。在顺序地址的选择中，加入了多路选择器和比较器判断下一个地址是不是 28，如果是 28，则选择地址为 0 作为下一个指令地址。

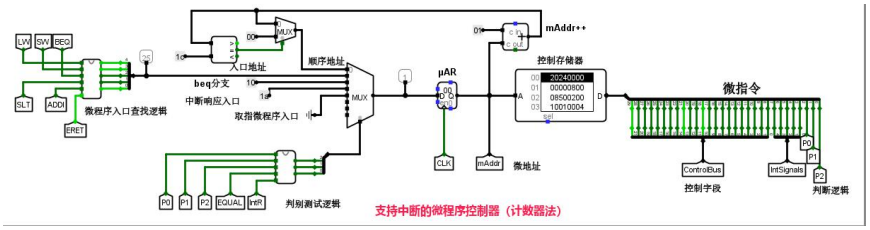


图 3.4 支持中断的微程序控制器

3.2.4 支持中断的微程序单总线 CPU

在文件 6.CPU 设计实验中，找到文件汇编工具及测试用例 4.4，里面有 Mars4_5.jar，双击打开，载入文件 sort-5-int.asm，根据注释选中菜单栏中的 Settings 的最下方的 Memory Configuration，设置更改为 Compact，data at address 0，完成后选中菜单栏中的 Run 的 Assemble，在 Execute 中可以看见指令的 Address，显示的两个中断程序入口指令(addi $sp,$sp,8)的地址 0x000030a4 和 0x000030ec 即是所求。若有判断是 1 号中断还是 2 号中断，选择对应的入口地址后传递到总线。如图 3.4。

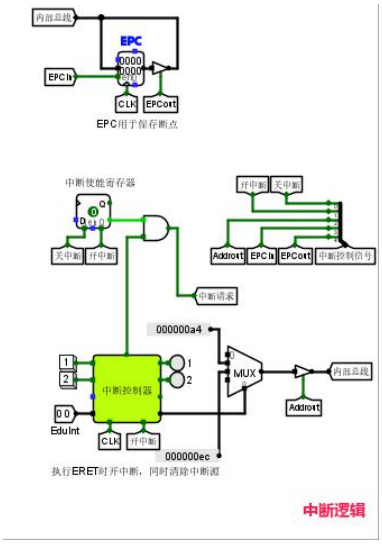


图 3.5 支持中断的微程序 CPU 中断电路

然后导入镜像文件即可。中断逻辑主要增加了对于当前 pc 的寄存器 EPC，以及中断信号控制器和中断信号选择器，当收到中断指令的时候 pc 进入 EPC 寄存器中对应着 EPCin，然后进行关中断操作，然后通过中断控制器输出选择的中断信号，通过多路选择器将对应的中断入口地址送入总线中。请求结束后，要通过 EPCout 信号将

上次的现场恢复。

3.2.5 支持中断的现代时序硬布线控制器状态机

根据状态转移图图 3.1，将每个状态对应的下一个次态填入真值表中，真值表如图 3.6 所示。其中到 26 的中断程序的转移需要信号 IR 才可以转移，相当于出现了中断信号。其余的如取址指令完之后需要相应的指令信号出现才会转移到相应的状态，如到状态 15 之后需要出现 equal 信号才会进入状态 16，否则要到判断中断的地方。

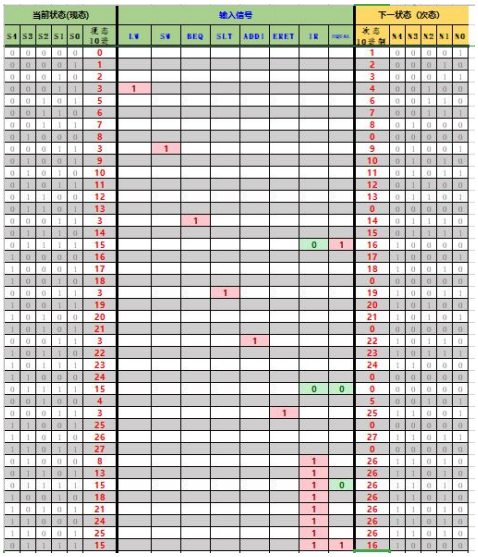


图 3.6 支持中断的现代时序硬布线控制器状态机

3.2.6 支持中断的现代时序硬布线控制器设计

根据所给的设计原理图，状态机将次态传给状态寄存器，状态寄存器将得到的现态传给状态机，其中对于微指令的最后三位分别为 P1、P2、P3，倒数四到八位则为中断信号，这点可根据微指令的构成得到。图 3.6 即为硬布线控制器的电路图。

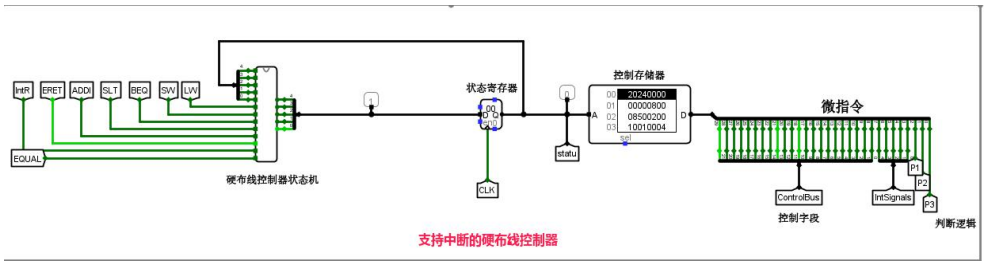


图 3.6 支持中断的现代时序硬布线控制器电路

3.3 实验步骤

1） 按照题目要求对于整个系统的架构进行分析，将 cpu 分成多个部件然后组装到一起，根据每一关构造各个模块，按照平台实验要求和老师对应的视频文件，利用对应的 excel 表完成对应的 circ 电路文件，并在每个模块构建完成后对其进行单元测试，以保证各个模块独自工作时功能是正常的。

2） 将完成的 circ 电路图用记事本打开，将代码复制后放入平台测试每一关。

3） 对应平台中测试机不同的测试集报错信息对自己的电路图进行修改。 在构建好的 CPU 上加载程序并运行，对于执行错误的地方进行单步调试并分析错误原因，然后对于电路进行修改，直到没有任何错误出现。

3.4 故障与调试

在写支持中断的微程序条件判别逻辑时，没有搞明白 P1、P2、P3 的优先级，导致写了好几次都没有通过，实际上应该为 P0 为 1 时，P1、P2 应该都为 0；P1 为 1 时，要先对 P1 进行判断，P2 为 1 时，要先看 P2 是否为 0。

3.5 测试与分析

对于含有中断的 CPU 测试：在 MEM 中加载 sort-5-int.hex 程序，ctrl+k 自动运行电路结果如图 3.7 所示。由运行结果可知该车工序运行至 0x7c8 节拍停下，指令计数为 252 符合实验要求完成有符号数的降序排序，排序结果如图 3.8 所示。

图 3.7 电路测试结果

000	23bd0400	2010ffff	20110000	ae300200	22100001	22310004	ae300200	22100001	22310004	ae300200	22100001	22310004	ae300200	22100001	22310004	ae300200
010	22100001	22310004	ae300200	22100001	22310004	ae300200	22100001	22310004	ae300200	20100000	2011001c	8e130200	8e340200	0274402a	11000002	ae330200
020	ae140200	2231fffc	12110001	1000fff7	22100004	2011001c	12110001	1000fff3	1000ffff	23bd0008	afb00000	afb10004	20310240	8e300000	22100001	ae300000
030	ae300004	ae300008	ae30000c	ae300010	ae300014	ae300018	ae30001c	8fb10004	8fb00000	23bdfff8	42000018	23bd0008	afb00000	afb10004	20310280	8e300000
040	2210ffff	ae300000	ae300004	ae300008	ae30000c	ae300010	ae300014	ae300018	ae30001c	8fb10004	8fb00000	23bdfff8	42000018	00000000	00000000	00000000
050	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
060	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
070	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
080	00000006	00000005	00000004	00000003	00000002	00000001	00000000	ffffffff	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000

图 3.8 MEM 中排序结果

4 总结与心得

4.1 实验总结

本次实验主要完成了如下几点工作：

1） 完成单总线 CPU 定长指令周期三级时序电路的设计，完成单总线 CPU 微程序以及硬布线电路的设计，完成现代时序中断机制的实现。

2） 实现了单总线 CPU 能按要求完成工作，通过测试可以得到正确的测试结果。实现了 CPU 对中断的处理，可以准确地完成中断。

4.2 实验心得

1） 通过本次的实验，主要对 CPU 运行的工作机制有了一定的了解，对 CPU 基本上由哪些部件构成，这些部件在其中又起着什么样的作用有了比较全面而且清晰的了解。除此之外，通过实际对电路图上手设计，能让这些原理和逻辑更好地更深刻地进入自己的知识体系。

2） 对于该实验课程，不得不说，该实验课程的慕课还有计算机组成原理的慕课都做得很好，没有冗余的内容，基本上可以很快地学习和了解相关内容，并根据老师所给视频和逻辑图理解原理，完成实验。

3） 对于该实验课的建议的话，主要是设计微指令的时候，老师视频和 PPT 里面给的指令并不全，然后实验中又出现了该指令，只能看着测试集找对应的十六进制串，然后来看这个指令究竟是什么。

4） 总的来说，该课程给我带来的收获还是很大的，理解原理才是真理。

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： 李田田

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： _____