第1章　MIPS 处理器设计实验

1.1 MIPS 单周期处理器设计实验

1.1.1 实验目的

学生掌握控制器设计的基本原理，能利用硬布线控制器的设计原理在 Logisim 平台中设计实现 MIPS 单周期 CPU。

1.1.2 实验内容

利用运算器实验，存储系统实验中构建的运算器、寄存器文件、存储系统等部件以及 Logisim 中其它功能部件构建一个 32 位 MIPS CPU 单周期处理器。

方案一：支持表 1.1 中的 8 条 MIPS 核心指令，最终设计实现的 MIPS 处理器能运行实验包中的冒泡排序测试程序 sort.asm，该程序自动在数据存储器 0~15 号字单元中写入 16 个数据，然后利用冒泡排序将数据升序排序，要求统计指令条数并与 MARS 中的指令统计数目进行对比。

表 1.1　核心指令集

#	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

1.1.3 注意事项

1）MIPS 虚存模式设置

由于实验中设计的单周期 MIPS 处理器并不包括内存管理单元 MMU 部件，所以程序运行时的访存地址均是物理地址，设计时采用指令存储器和数据存储器分离的哈佛架构。访问数据时应该访问数据存储器，数据存储器的地址起始地址是 0，实验包中的测试程序均直接使用了

0 号地址开始的数据单元，为了使测试程序在 MARS 和实验设计的处理器中的运行结果保持一致，必须配置 MARS 模拟器中的 MIPS 虚存模式，具体可以选择菜单 Setting 的 Memory Configration 选项，该选项可以设置 MIPS 虚拟地址空间的地址模式，这里应将虚存模式设置为 Compact 模式，这样数据段起始位置 0 开始的位置，如图 1.1 所示。注意，如果采用 Default 模式在 MARS 中运行 sort.asm 排序程序时访存指令会越界访问代码段或内核段，引起保护错。

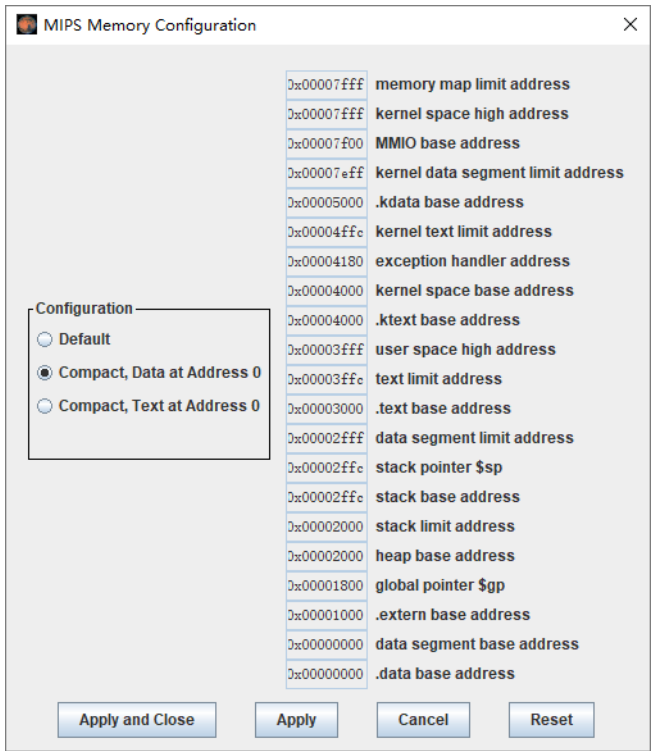

图 1.1 MIPS 虚存模式配置

2）MIPS 机器指令导出

在 MARS 中可以利用 File 菜单中的 Dump Memory 功能将汇编程序的代码段的机器指令和数据段的数据导出，为了能直接在 Logisim 的 RAM 和 ROM 组件中宋使用，推荐采用十六进制文本的方式导出到某个文本文件，然后在文本文件第一行加入“v2.0 raw”，这样导出的文件即可直接加载到 Logisim 平台的 ROM 和 RAM 组件中，如图 1.2 所示。

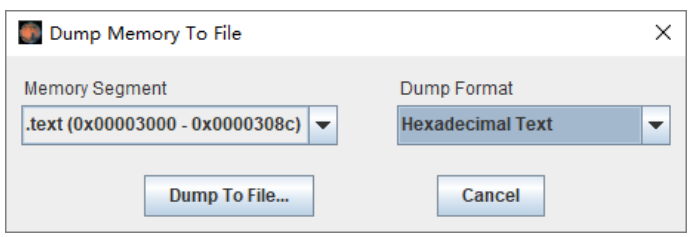

图 1.2 MIPS 代码导出

4、MIPS 寄存器文件库

存储系统实验中我们仅仅设计了包含 4 个寄存器的寄存器文件，为满足 MIPS 单周期处理

器设计的需要，这里我们提供了一个包含 32 个寄存器的 MIPS 寄存器文件的库 CS3410.jar，该库由美国康奈尔大学开发，在 Logisim 平台中可以通过加载 JAR 库的方式加载第三方 JAVA 库，如图 1.3 所示。

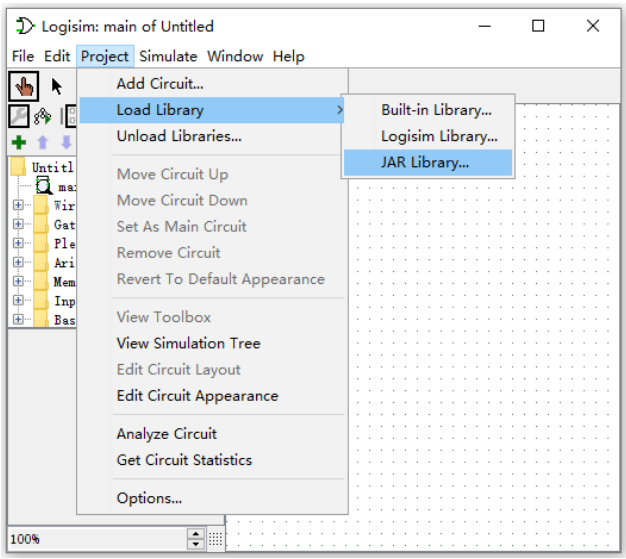

图 1.3 JAR 库加载

选择实验包中的 CS3410.jar 库，加载成功后 logisim 左下角的组件库会增加 CS3410 组件的选项，其中第一个组件就是寄存器文件，添加该组件到画布中，如图 1.4 所示，这个寄存器文件的引脚 rA，rB 为读寄存器编号，rW 为写入寄存器编号，WE 为写使能，W 为写入数据，A、B 为 rA，rB 寄存器的输出值，该组件直接提供了 32 个寄存器观察窗口，非常直观，用户还可以使用手型的戳工具直接修改寄存器的值。需要注意的该组件缩小显示时组建上数字的显示可能不正常。

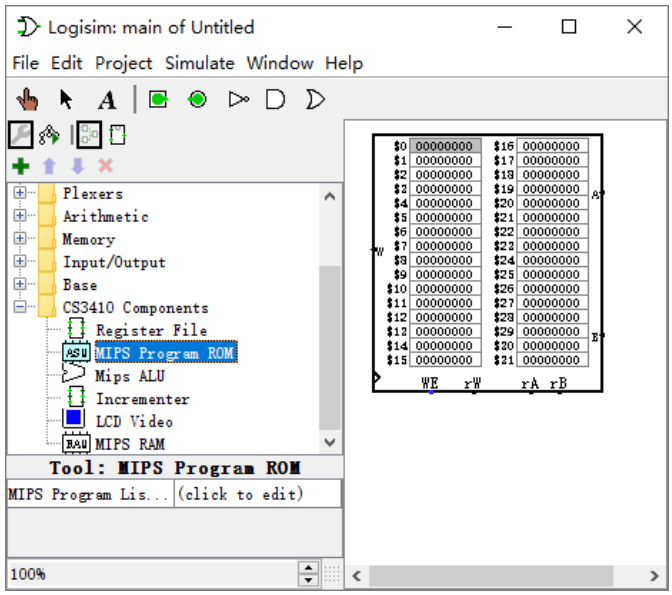

图 1.4 MIPS 寄存器文件库

1.1.4 实验思考

1.2 MIPS 多周期处理器设计实验

1.2.1 实验目的

学生理解 MIPS 多周期处理器的基本原理，能利用硬布线控制器的设计原理设计实现 MIPS 多周期 CPU。

1.2.2 实验内容

构造多周期 MIPS 处理器，要求能支持表 1.1 中的 8 条 MIPS 核心指令，最终设计实现的 MIPS 处理器能运行实验包中的冒泡排序测试程序 sort.asm，该程序自动在数据存储器 0~15 号字单元中写入 16 个数据，然后利用冒泡排序将数据升序排序，实验电路应能自动统计指令数目、时钟周期数。

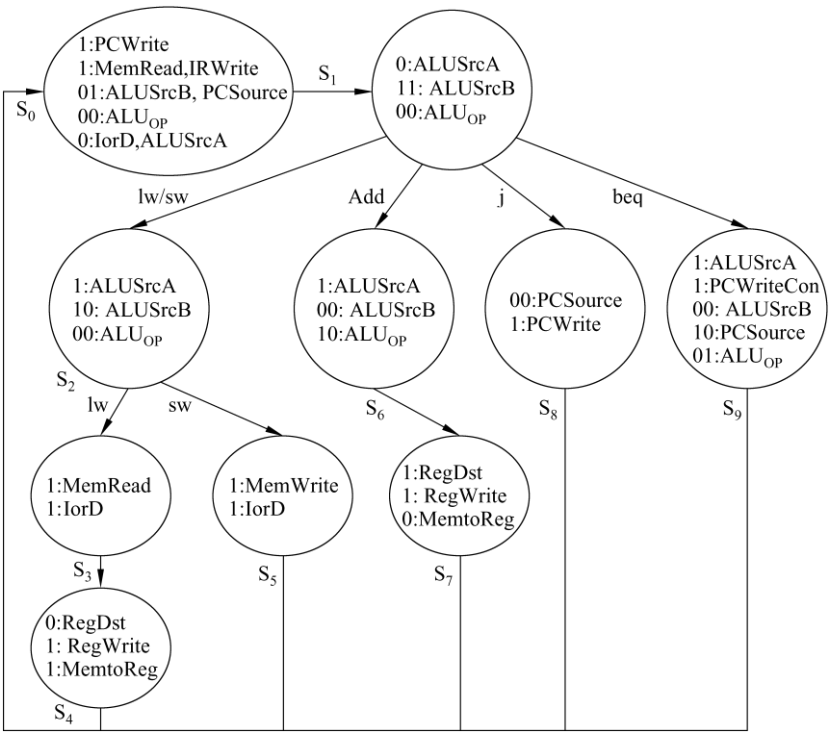

图 1.5 指令执行状态转换图

1.2.3 实验步骤

1）构建主要功能部件和数据通路

在 Logisim 平台中设计 MIPS 多周期处理器所需的主要功能部件，其中寄存器文件可以参考本章 1.1.3 节的介绍的标准库，其中运算器既可以使用运算器实验中自行设计的运算器，也可以使用标准库中的 ALU 模块，对照图 5.13 所示多周期 MIPS 处理器数据通路，最终将各功能部件连接形成数据通路。

2）构建硬布线控制器

列出上述 8 条指令的 Moore 型状态转换图，如图 1.5 所示。这里仅仅给出了 5 条基本指令的状态转换图，其它 3 条指令请酌情补充完善。图中 10 个状态的意义分别为：**S_0** 为取指、**S_1** 为译码及取操作数、**S_2** 为访存指令计算有效地址、**S_3** 为 lw 指令访问存储器、**S_4** 为 lw 指令数据写回寄存器、**S_5** 为 sw 指令数据写回存储器、**S_6** 为 Add 指令两数加运算、**S_7** 为 Add 指令数据写回寄存器、**S_8** 为 beq 指令目标地址送 PC、**S_9** 为 j 指令目标地址送 PC。10 个状态需要四个寄存器来表示，图中 S 的编号用对应的四位二进制编码表示，状态之间的转换关系如表 1.2 所示。

表 1.2 控制器状态状态表

现态	输入(指令操作码)/次态						输出(即操作控制信号)
	XXXX	Add	lw	sw	beq	j	
0000(S0)	0001						MemRead =IRWrite= PCWrite=1， IorD=0，ALUSrcA =0, ALUSrcB =01, PCsource=01，ALUop =00
0001(S1)		0110	0010	0010	1000	1001	ALUSrcA =0 , ALUSrcB=11, ALUop =00
0010(S2)			0011	0101			ALUSrcA=1 ,ALUSrcB=10,ALUop=00
0011(S3)		0100					MemRead=1 ,IorD=1
0100(S4)	0000						RegDst=0, RegWrite=1,MemtoReg=1
0101(S5)	0000						MemWrite=1, IorD=1
0110(S6)		0111					ALUSrcA=1, ALUSrcB=00,ALUop =10
0111(S7)	0000						MemtoReg=0,RegWrite= RegDst=1
1000(S8)	0000						PCSource=00,PCWrite=1
1001(S9)	0000						ALUSrcA=1 ,PCWriteCon=1, ALUSrcB=00,PCSource=10,ALUop=01

利用译码电路生成指令译码信号 add、lw、sw、beq、j 等 8 条指令的译码信号，根据次态和现态之间的逻辑关系，利用逻辑表达式生成状态转换电路，根据控制信号与现态之间的关系，给出对应的逻辑表达式，并给出最终的逻辑电路，将状态寄存器、状态转换电路和控制信号电路封装在一起就是最终的多周期处理器硬布线控制器单元。

3）系统调试

将控制器输出与所有控点相连，构造完整的多周期 MIPS 处理器电路，根据状态转换图逐条指令测试控制器逻辑，状态转换无误后测试 sort.asm 程序直至功能正确。注意多周期处理器中指令和数据存放在同一存储器中，sort.asm 中使用的数据段在主存 0~15 号单元，所以代码段不能放在低地址部分，实验时应注意程序代码如何存放。

1.2.4 实验思考

1）利用 mars 统计 sort.asm 程序运行中执行指令总数（单周期处理器周期数），和实现的多周期处理器周期数进行对比，多周期处理器运行 sort.asm 程序周期数明显增多，是否多周期处理器性能更差？

1.3 微程序控制器设计实验

1.3.1 实验目的

学生掌握微程序控制器设计的基本原理，能利用微程序控制器的设计原理设计实现多周期 MIPS 处理器。

1.3.2 实验内容

对照**错误!未找到引用源。**所示多周期 MIPS 处理器数据通路，采用微程序控制器的设计

方法实现控制器，构造多周期 MIPS 处理器，要求能支持表 1.1 中的 8 条 MIPS 核心指令，最终设计实现的 MIPS 处理器能运行实验包中的冒泡排序测试程序 sort.asm，该程序自动在数据存储器 0~15 号字单元中写入 16 个数据，然后利用冒泡排序将数据升序排序。实验电路应能自动统计指令数目、时钟周期数。

1.3.3 实验步骤

1） 构建主要功能部件和数据通路

在 Logisim 平台中设计 MIPS 多周期处理器所需的主要功能部件，其中寄存器文件可以参考本章 1.1.3 节的介绍的标准库，其中运算器既可以使用运算器实验中自行设计的运算器，也可以使用标准库中的 ALU 模块，对照图 5.13 所示多周期 MIPS 处理器数据通路，最终将各功能部件连接形成数据通路。

2） 设计微指令

结合微程序控制器设计基本原理设计 MIPS 多周期处理器微指令格式，结合图 1.5 中的状态转换图，为每一条 MIPS 指令构建微程序，微指令顺序控制方法可自行选择。

3） 构建微程序控制器

设计地址转移组合逻辑，构建微程序控制器,注意这里由于采用了专门的控制器存储器，所以微命令寄存器并不是必须的，可以直接将控制存储器读出的微指令送地址转移逻辑进行解析，最后将微程序控制器进行封装。

4） 系统调试

将控制器输出与所有控点相连，构造完整的多周期 MIPS 处理器电路，根据状态转换图逐条指令测试控制器逻辑，状态转换无误后测试 sort.asm 程序直至功能正确。注意多周期处理器中指令和数据存放在同一存储器中，sort.asm 中使用的数据段在主存 0~15 号单元，所以代码段不能放在低地址部分，实验时应注意程序代码如何存放。

1.3.4 实验思考

1) 对比实验中的微程序控制器和硬布线控制器设计，哪个更简单，为什么？