***2020***



**计算机组成原理 ·实验报告·**

j0242087[1]

|  |  |
| --- | --- |
| 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](mailto:1359947339@qq.com) |
| 完成日期： | 2020-12-10 |



目 录

[1 CPU设计实验 2](#_Toc58918274)

[1.1 设计要求 2](#_Toc58918275)

[1.2 方案设计 2](#_Toc58918276)

[1.3 实验步骤 5](#_Toc58918277)

[1.4 故障与调试 14](#_Toc58918278)

[1.5 测试与分析 15](#_Toc58918279)

[2 总结与心得 18](#_Toc58918280)

[2.1 实验总结 18](#_Toc58918281)

[2.2 实验心得 18](#_Toc58918282)

[参考文献 20](#_Toc58918283)

# CPU设计实验

## 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 |

## 方案设计

### 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。

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

### 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；
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

### 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；
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

## 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：

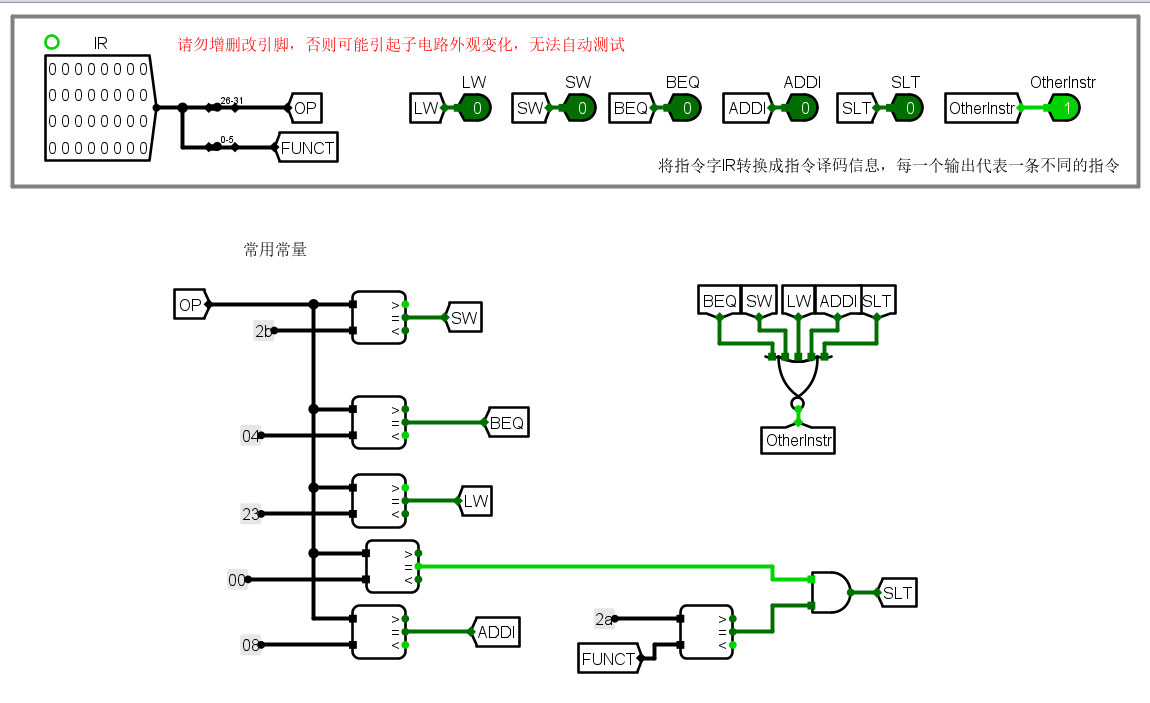


图1-1 指令译码器

1. 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：

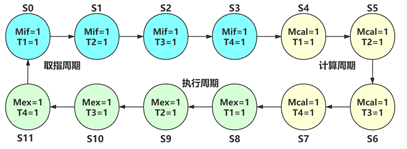


图1-2 定长指令周期状态图

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。



图1-3 3级时序状态转换表

1. 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：

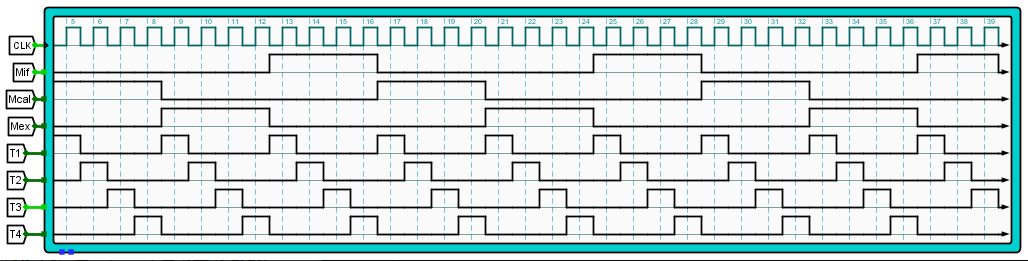


图1-4 状态周期电位和节拍电位信号

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。

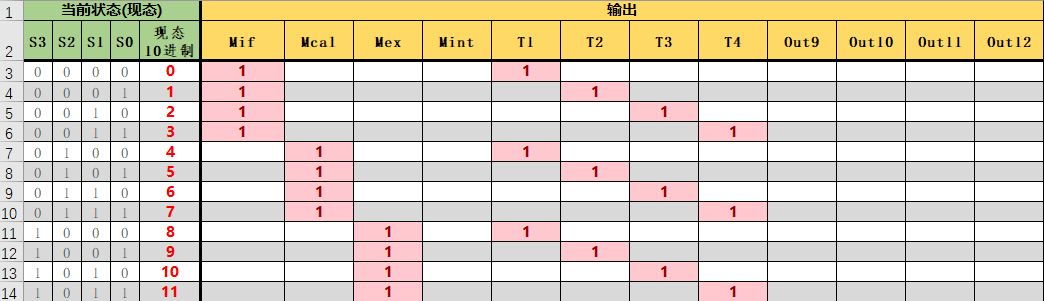


图1-5 输出函数真值表

1. 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：



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。

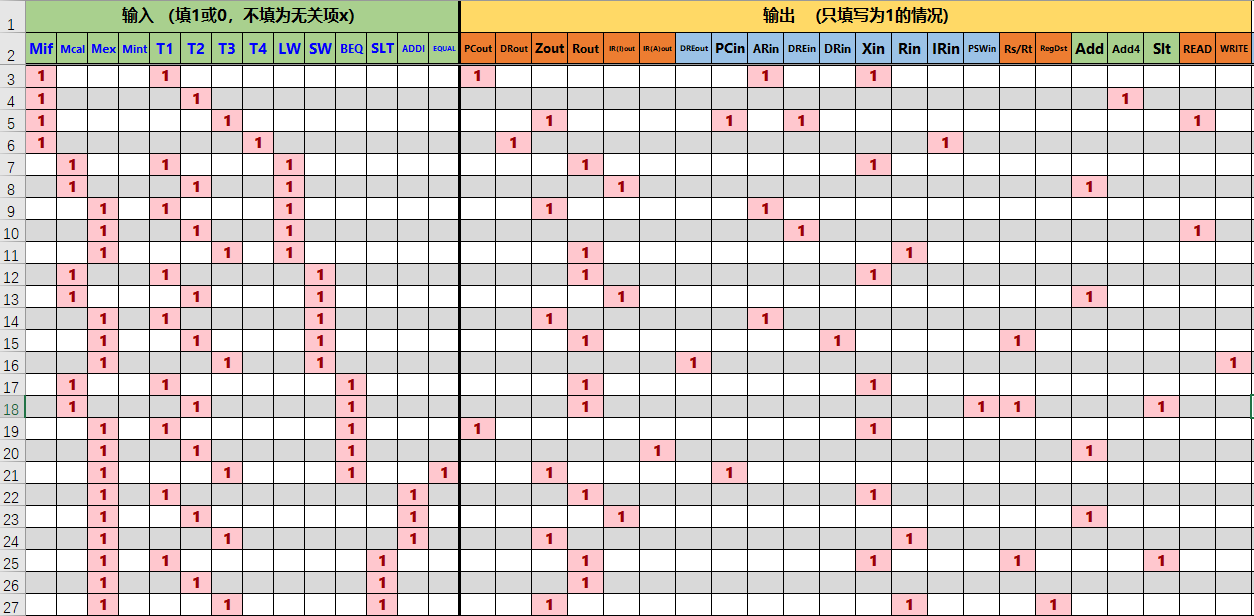


图1-6 组合逻辑真值表

1. 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。

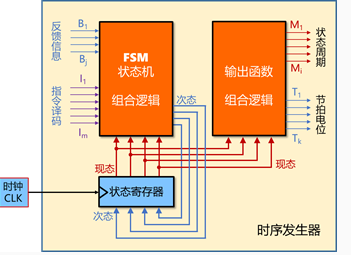


图1-7 3级时序发生器框架

连接好的硬布线控制器（3级时序）如图1-8所示：

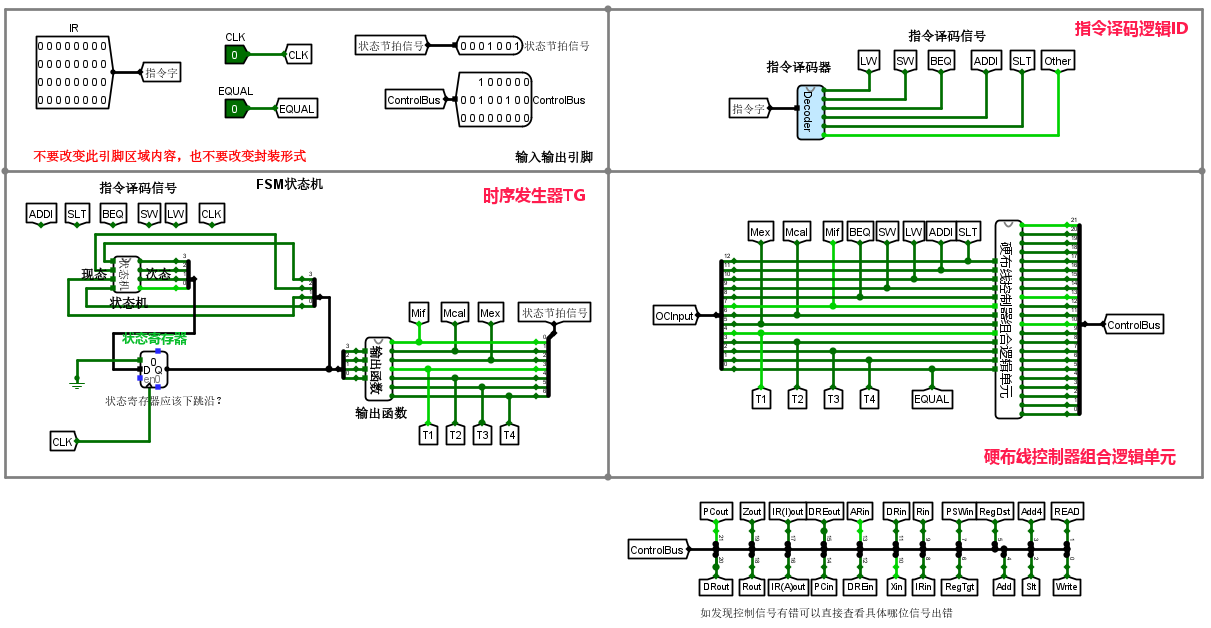


图1-8 硬布线控制器（3级时序）

1. 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：

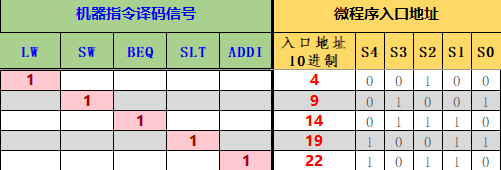
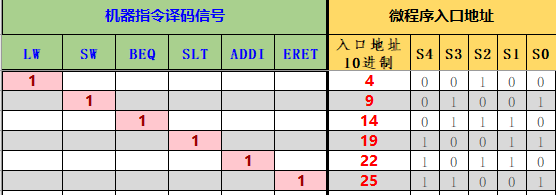
 

图1-9 微程序入口地址（现代时序） 图1-10 微程序地址入口（中断）

1. 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：

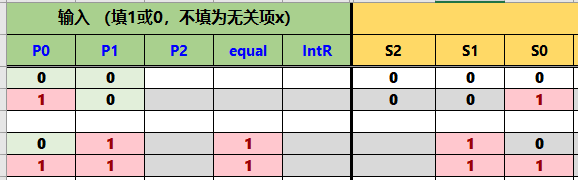


图1-11 条件判别测试逻辑（现代时序）

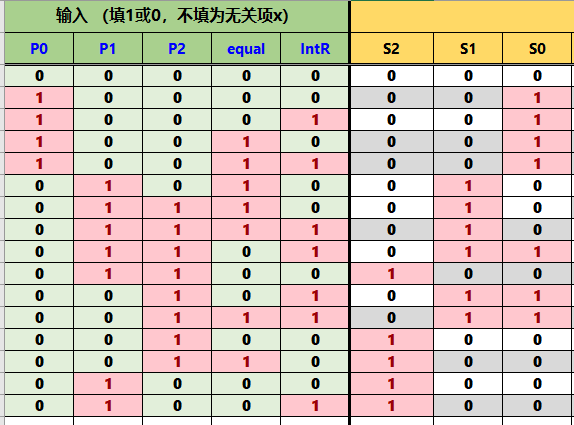


图 1-12 条件判别测试逻辑（中断实现）

1. 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：

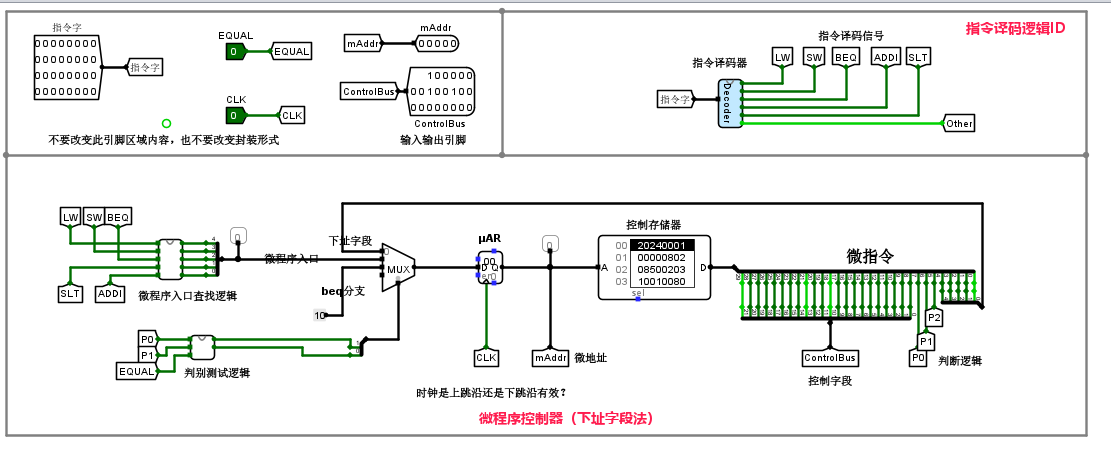


图1-13 微程序控制器（现代时序）

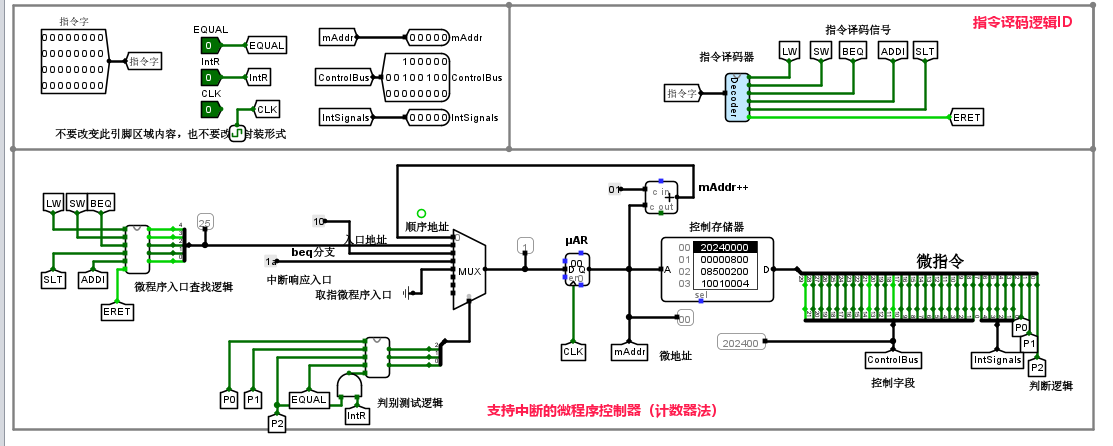


图1-14 支持中断的微程序控制器

1. 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。

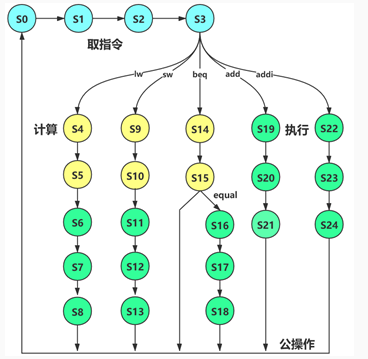
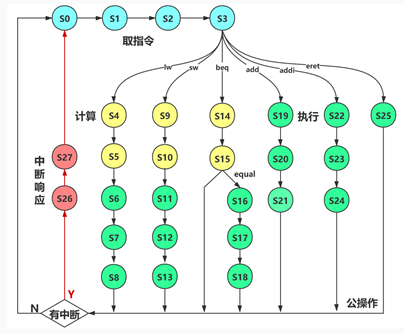
 

图1-15 现代时序状态图 图 1-16 中断机制状态图

1. 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：

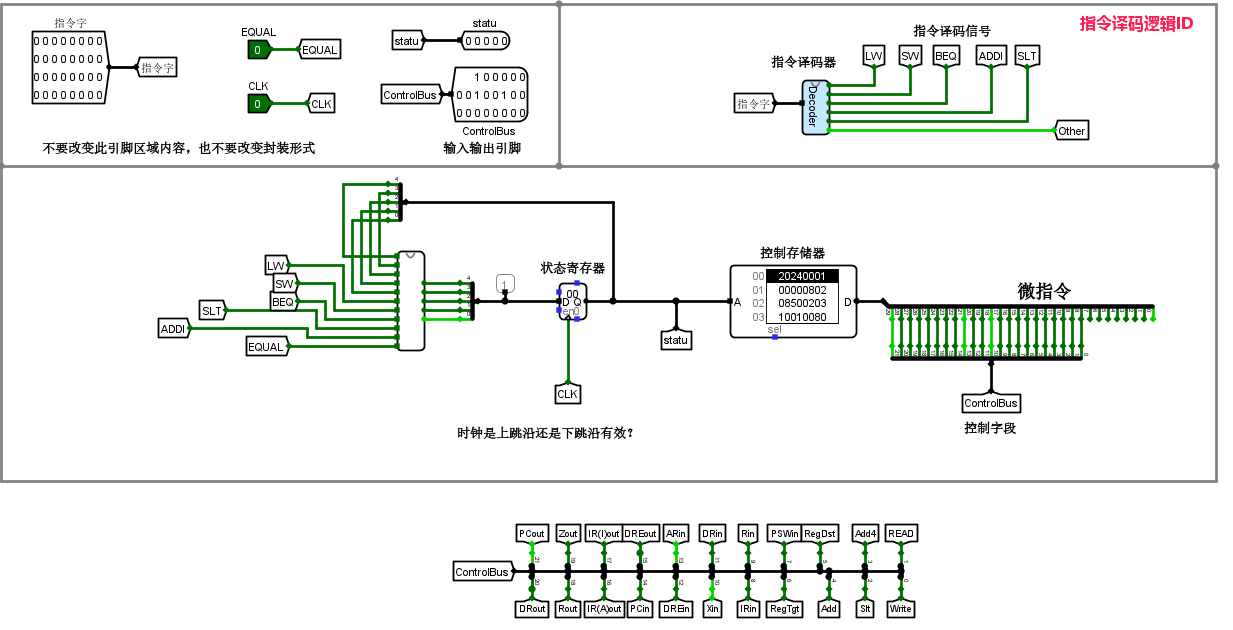


图1-17 硬布线控制器（现代时序）

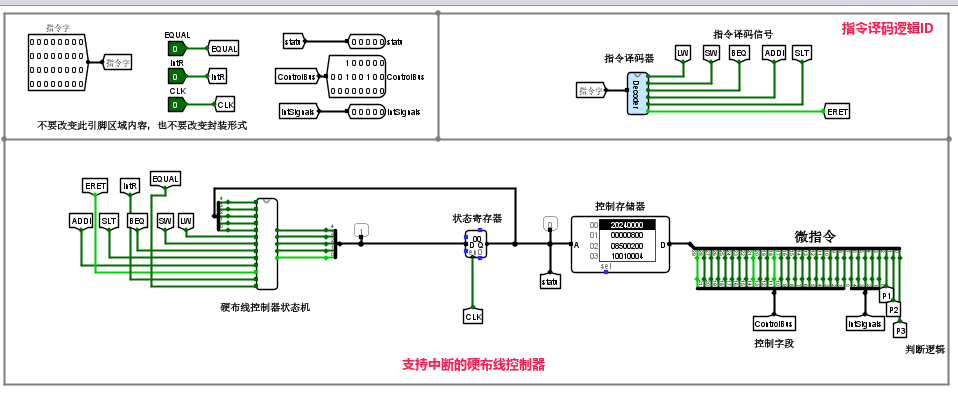


图 1-18 支持中断的硬布线控制器

1. 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：

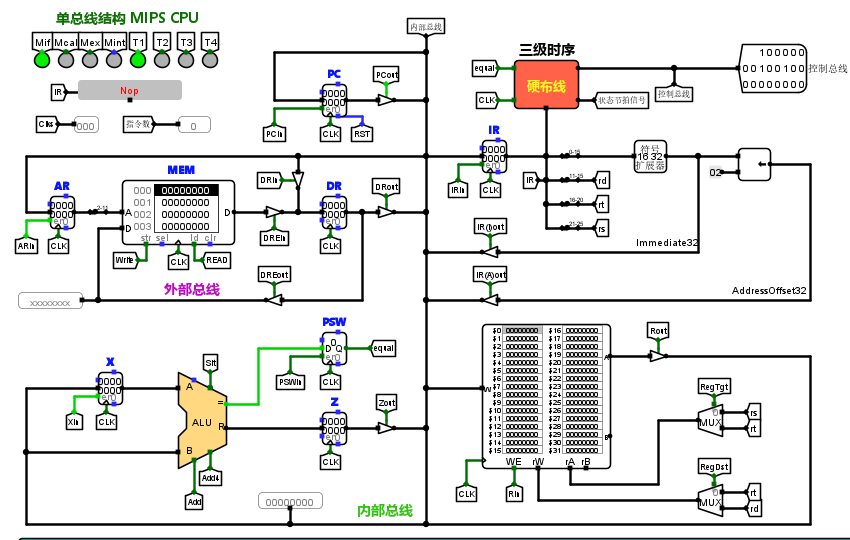


图 1-19 单总线CPU（定长指令周期3级时序）

1. 单总线CPU（现代时序）：与单总线CPU（定长指令3级时序）相比，现代时序单总线CPU的数据总线及布局完全一样，不同之处只在于控制信号的生成方式不同，单总线CPU（现代时序）的全貌如图1-20：

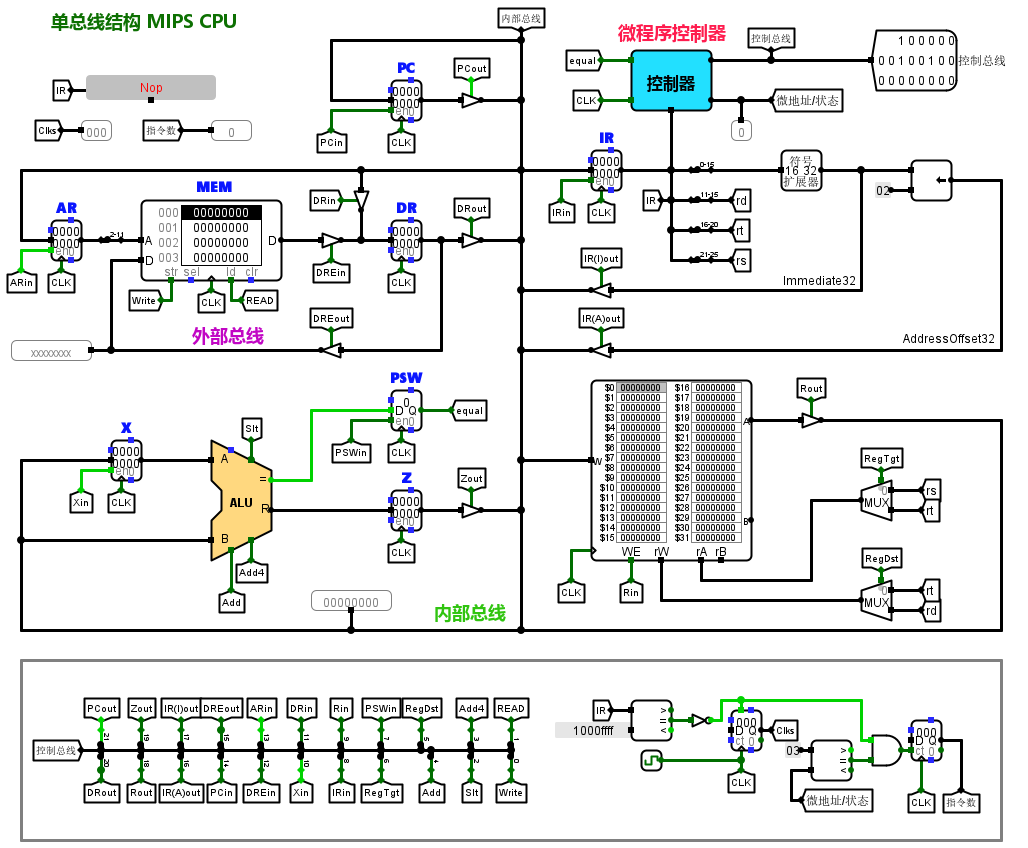


图 1-20 现代时序单总线CPU

1. 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：

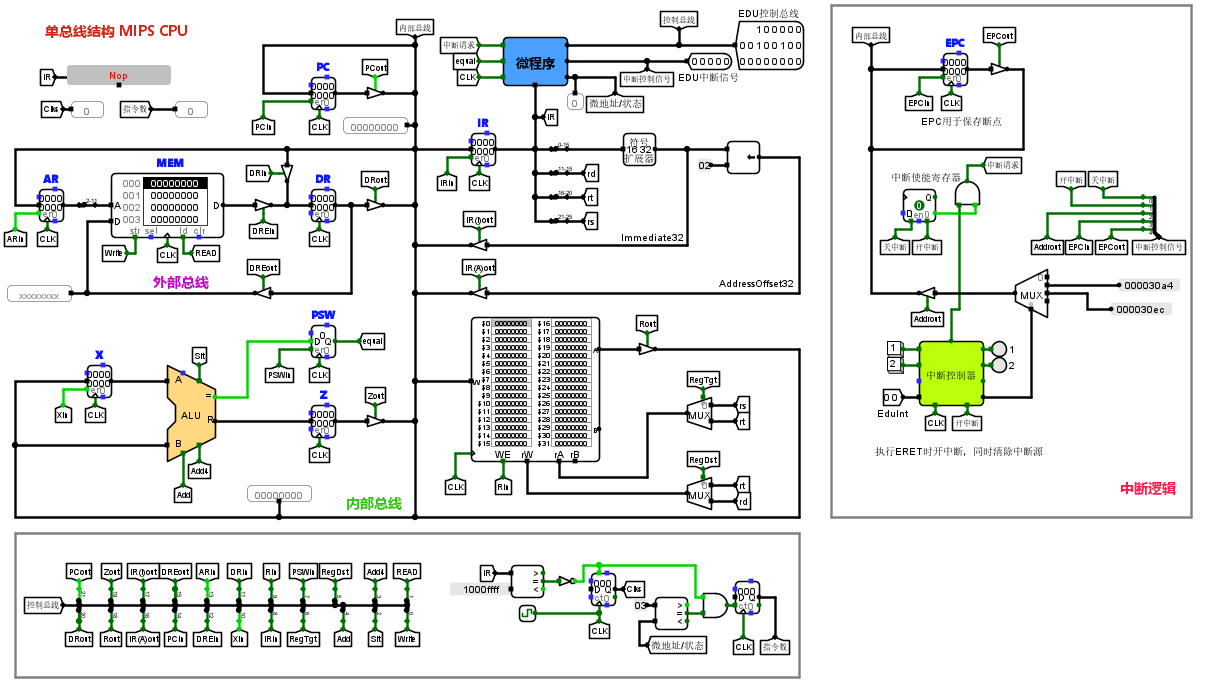


图 1-21 支持中断机制现代时序单总线CPU

## 故障与调试

### 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

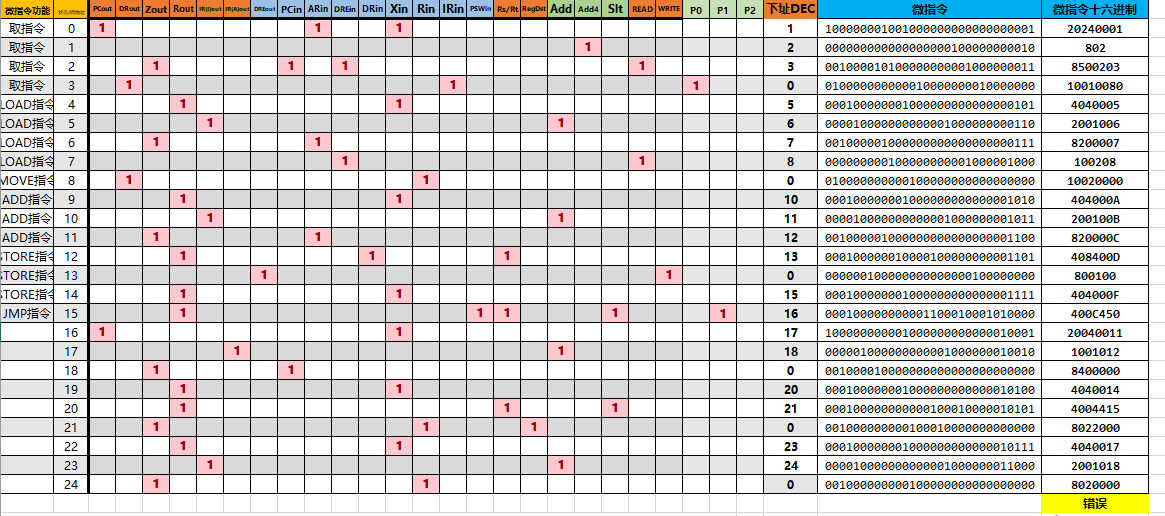


图 1-22 单总线现代时序微程序控制器设计图

**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。

## 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：

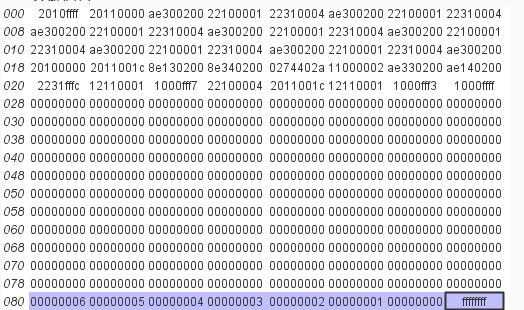


图 1-23 单总线CPU（3级时序）执行sort-5.hex后内存布局

（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：

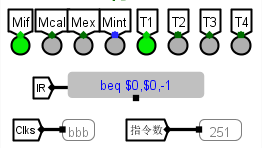


图 1-24 单总线CPU（3级时序）执行sort-5.hex所需时钟周期数

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：

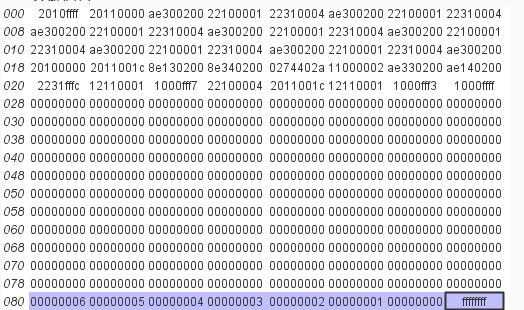


图 1-25 单总线CPU（现代时序）执行sort-5.hex后内存布局

（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：

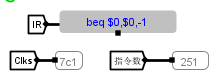


图 1-26 单总线CPU（现代时序）执行sort-5.hex所需时钟周期数

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：

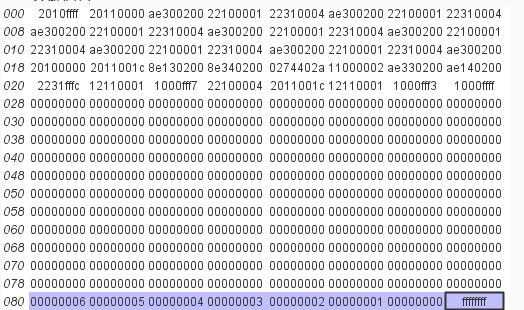


图 1-27 单总线CPU（现代时序）执行sort-5.hex后内存布局

（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：

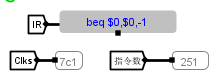


图 1-28 单总线CPU（现代时序）执行sort-5.hex所需时钟周期数

（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：

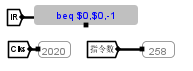
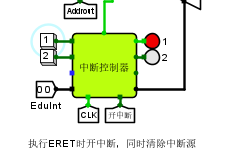
 

图1-29 中断申请是指令数 图1-30 中断申请1的信号灯变红

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：

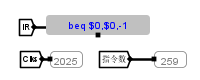


图1-31 执行完中断响应后指令数

# 总结与心得

## 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。

## 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。
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.

|  |
| --- |
| 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:** |