***2022***



**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： | CS2003班 |
| 学 号： | U202015375 |
| 姓 名： | 汪宇飞 |
| 电 话： | 15926575495 |
| 邮 件： | [2453253921@qq.com](mailto:2453253921@qq.com) |
| 完成日期： | 2022-07-05 |



目 录

[**1 CPU设计实验 2**](#_Toc108056914)

[**1.1 设计要求 2**](#_Toc108056915)

[**1.2 方案设计 2**](#_Toc108056916)

[1.2.1 MIPS指令译码器设计 2](#_Toc108056917)

[1.2.2 支持中断的微程序入口查找逻辑 3](#_Toc108056918)

[1.2.3 支持中断的微程序条件判别测试逻辑 4](#_Toc108056919)

[1.2.4 支持中断的微程序控制器设计 5](#_Toc108056920)

[1.2.5 支持中断的微程序单总线CPU设计 6](#_Toc108056921)

[1.2.6 支持中断的现代时序硬布线控制器状态机设计 7](#_Toc108056922)

[1.2.7 现代时序硬布线控制器设计 8](#_Toc108056923)

[**1.3 实验步骤 8**](#_Toc108056924)

[**1.4 故障与调试 8**](#_Toc108056925)

[1.4.1 硬布线状态机问题 8](#_Toc108056926)

[**1.5 测试与分析 9**](#_Toc108056927)

[**1.6 实验总结 10**](#_Toc108056928)

[**1.7 实验心得 10**](#_Toc108056929)

# CPU设计实验

## 设计要求

根据实验要求、使用所提供的circ文件和excel表格、利用logisim平台中现有运算部件构建一个支持中断的微程序控制器，要求支持lw、sw、beq、slt和addi指令，并且能够支持中断响应，指令功能如表1-1所示。

表1-1 需要实现的指令功能

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | MIPS指令 | RTL功能描述 | 说明 |
| lw | lw rt,imm(rs) | R[rt]←M[R[rs]+SignExt(imm)] |  |
| sw | sw rt,imm(rs) | M[R[rs]+SignExt(imm)] ← R[rt] |  |
| beq | beq rs,rt,imm | if(R[rs]=R[rt]) PC ←PC+SignExt(imm)<<2 |  |
| slt | slt rd,rs,rt | R[rd] ←R[rs]<R[rt] | 有符号比较 |
| addi | addi rt,rs,imm | R[rt] ←R[rs]+SignExt(imm) | 不考虑溢出 |
| eret | eret | PC ←EPC,IE ←1 |  |

基于educoder上的实验指引，设计硬布线控制器内部逻辑，实现支持中断的现代时序单总线结构的MIPS CPU，并且能够最终调试运行标准测试程序sort-5-int.hex。

## 方案设计

### 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，即使用多个比较器与输入信号MIPS指令字IR中的OP字段与各个指令所对应的OP相比较，相同则输出对应的输出信号。其中对于slt指令的判断需要在比较OP的同时也比较FUNCT字段。

根据分析并结合各个指令的OP，所设计得出的电路如图1-1所示。

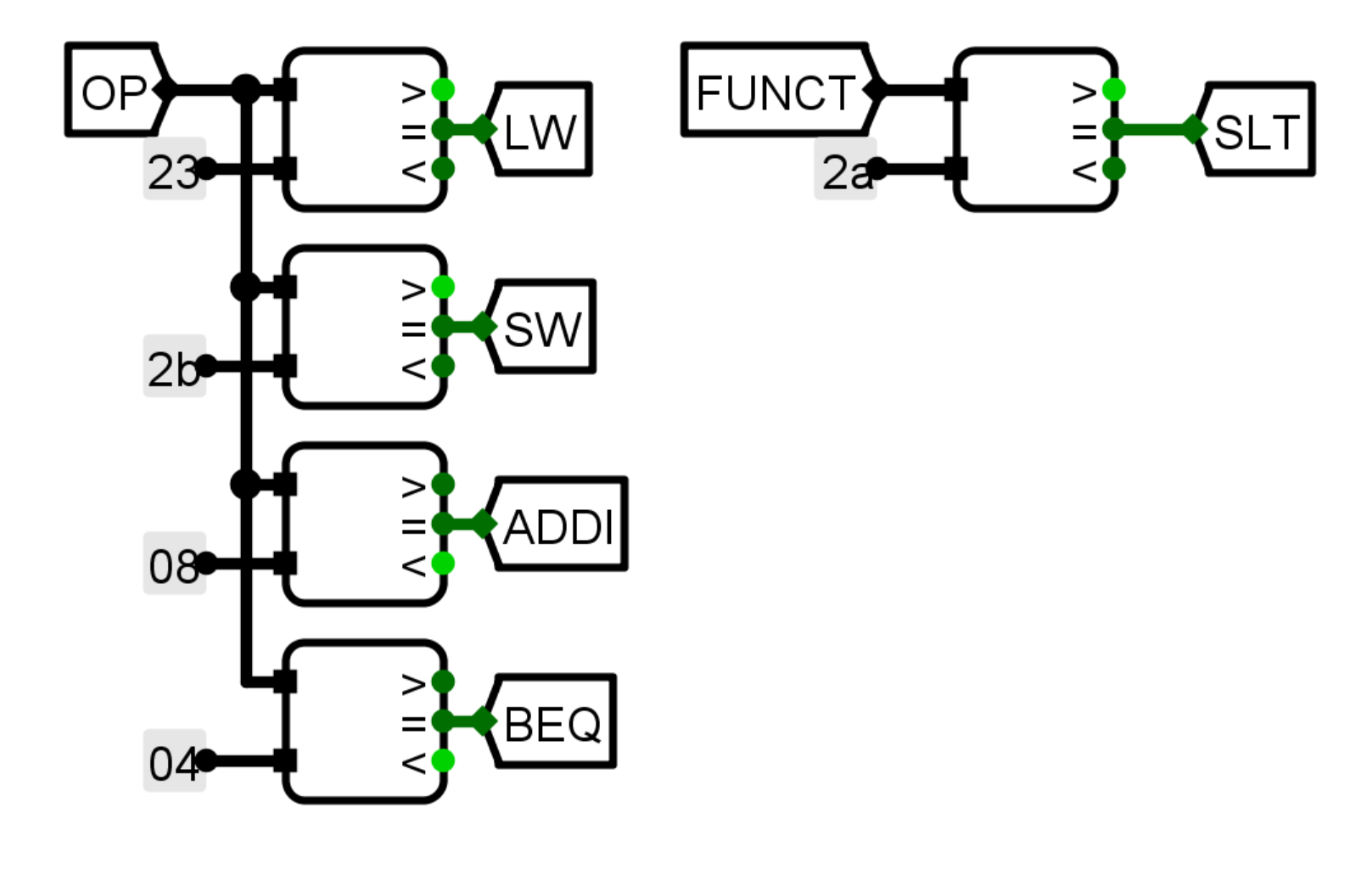


图1-1 MIPS指令译码器

### 支持中断的微程序入口查找逻辑

设计如图1-2所示的电路，根据指令译码信号生成5位的微程序入口地址。

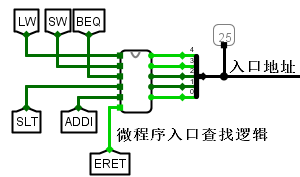


图1-2 微程序入口查找逻辑电路

地址转移逻辑如图1-3所示。

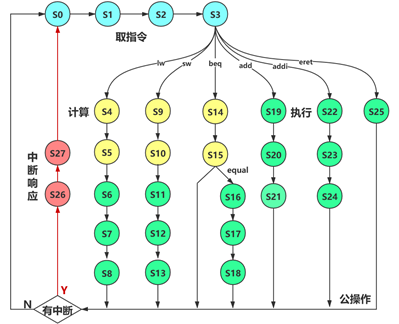


图1-3 地址转移逻辑

填写所提供的6号表格中的微程序入口地址表格，由图1-3中可以得出lw、sw、beq、slt、addi和eret指令的入口地址分别为4、9、14、19、22和25，填入表格如图1-4所示。



图1-4 微程序入口地址表格

将表格自动生成的表达式使用logisim的分析组合逻辑电路功能即可自动生成电路。

### 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，实现对应组合逻辑。

P0、P1和P2为输入判别测试位。P0为1时需要根据指令功能进行微程序分支， P1为1时需要根据equal标志进行微程序分支，P2为1时表示是当前微程序的最后一条微指令。

equal为输入条件状态位，为1表示运算相等。IntR为1时表示存在中断请求，需要转中断响应微程序执行。

根据上述分析填写所提供的4号表格中的组合逻辑真值表，如图1-5所示。

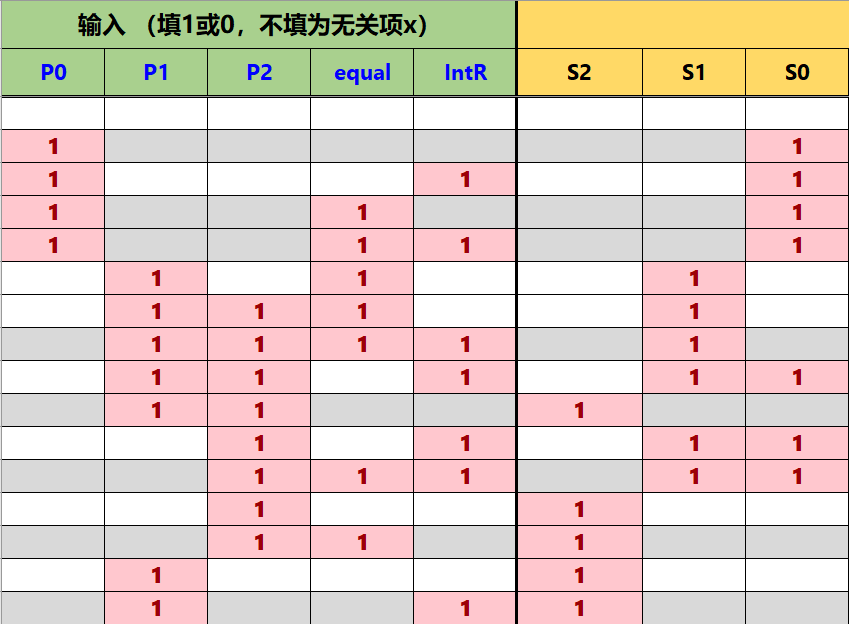


图1-5 组合逻辑真值表

将表格自动生成的表达式使用logisim的分析组合逻辑电路功能自动生成电路。

### 支持中断的微程序控制器设计

根据指令操作以及实验指引，填写6号表格中的微程序自动生成表，得出所对应的16进制微指令，如图1-6所示。

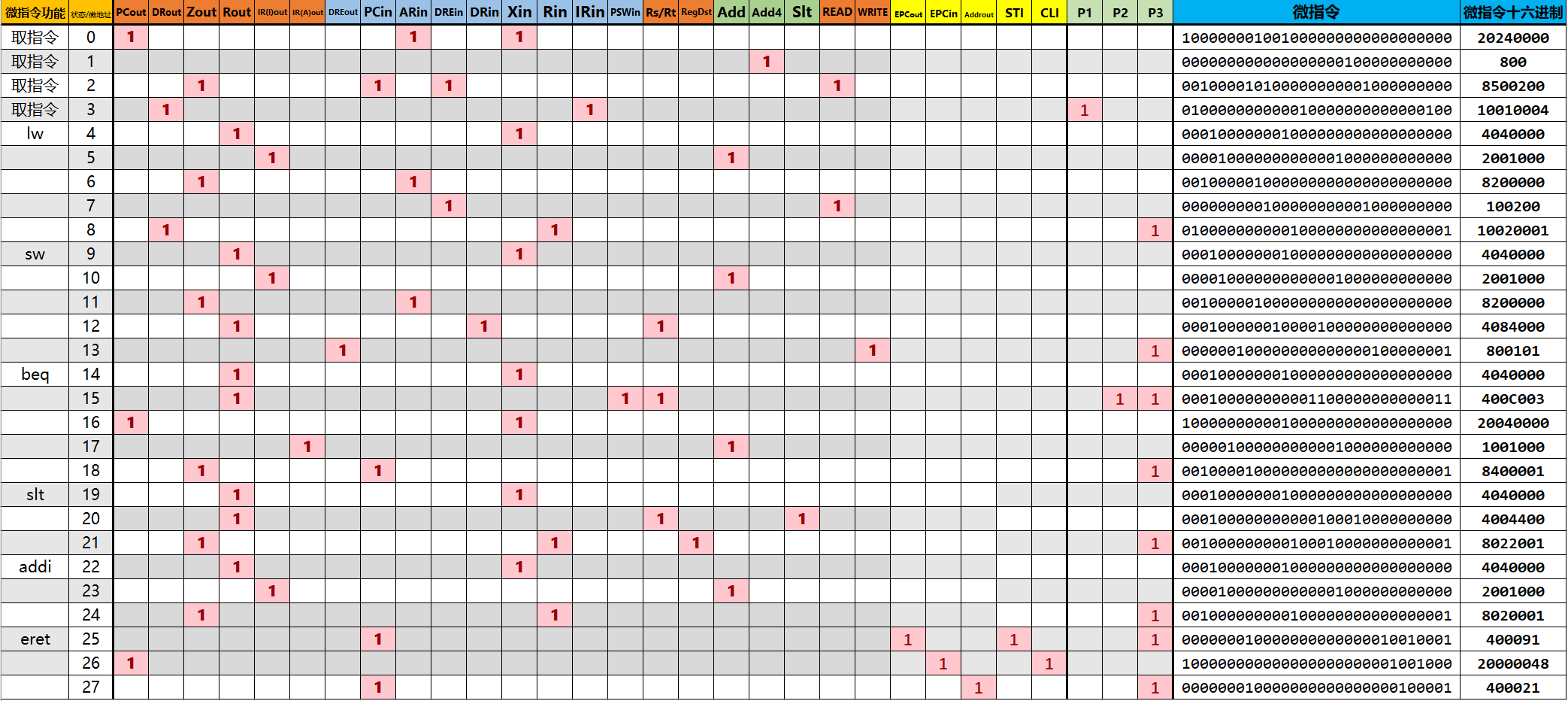


图1-6 微程序自动生成表

将所得到的微指令存入到对应的控制存储器中，如图1-7所示。

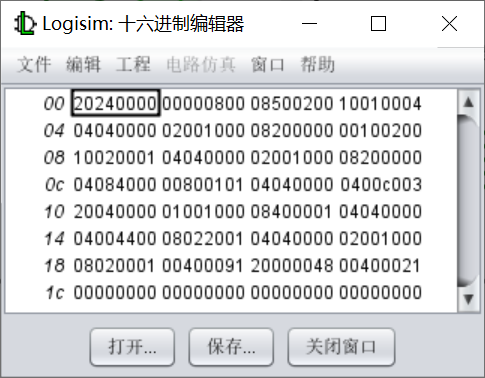


图1-7 控制存储器

将前面所设计的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，实现微程序控制器的主要数据通路。根据功能，判别测试逻辑所控制的多路选择器的前四个输入端从0至3分别为：顺序地址即下一状态、微程序入口查找逻辑的输出即微程序入口、beq分支equal为1时的0x10、中断响应入口的0x1a。

对于寄存器，应设置为下降沿有效。

根据分析所得电路如图1-8所示。

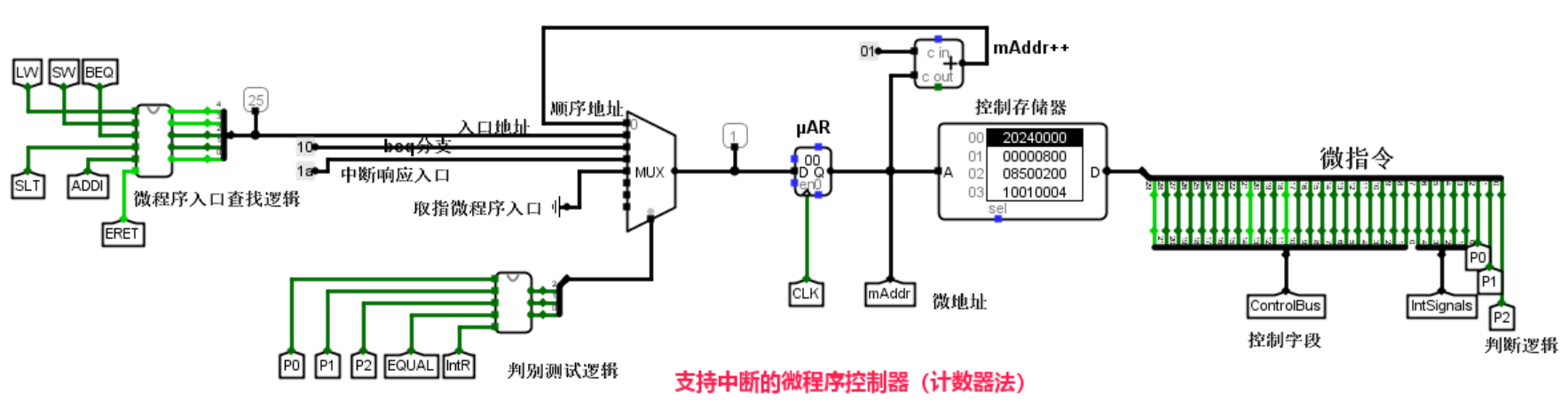


图1-8 支持中断的微程序控制器

### 支持中断的微程序单总线CPU设计

根据实验指引，打开Mars4\_5.jar文件，打开sort-5-int.asm文件，将Memory Configuration更改为Compact,data at address 0后进行Run中的Assemble，点击Execute查看。

在Text Segment部分查找中断程序入口指令addi $sp,$sp,8所对应的地址，一共有两个，分别为0x000030a4和0x000030ec，分别如图1-9和图1-10所示。

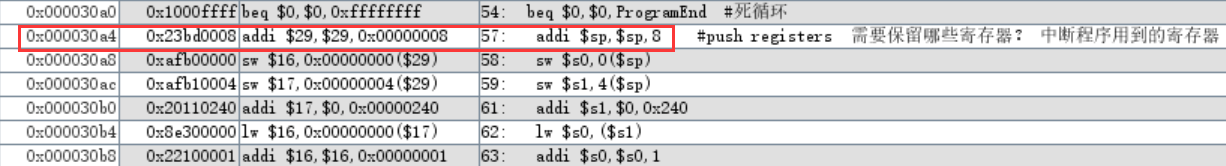


图1-9 0x000030a4

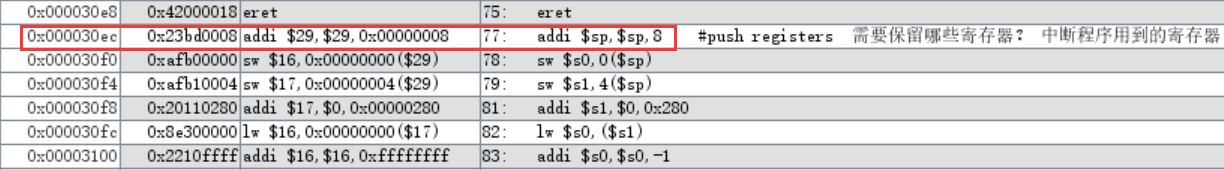


图1-10 0x000030ec

得出地址后可以设计中断逻辑，根据中断控制器的输出来选择相对应的地址。

对于保存断点的EPC，寄存器的输入和输出都连接内部总线，输入使用EPCin控制中断地址的输入，输出使用EPCout控制的三态门实现中断地址的恢复。

根据上述分析实现中断逻辑，电路图如图1-11所示。

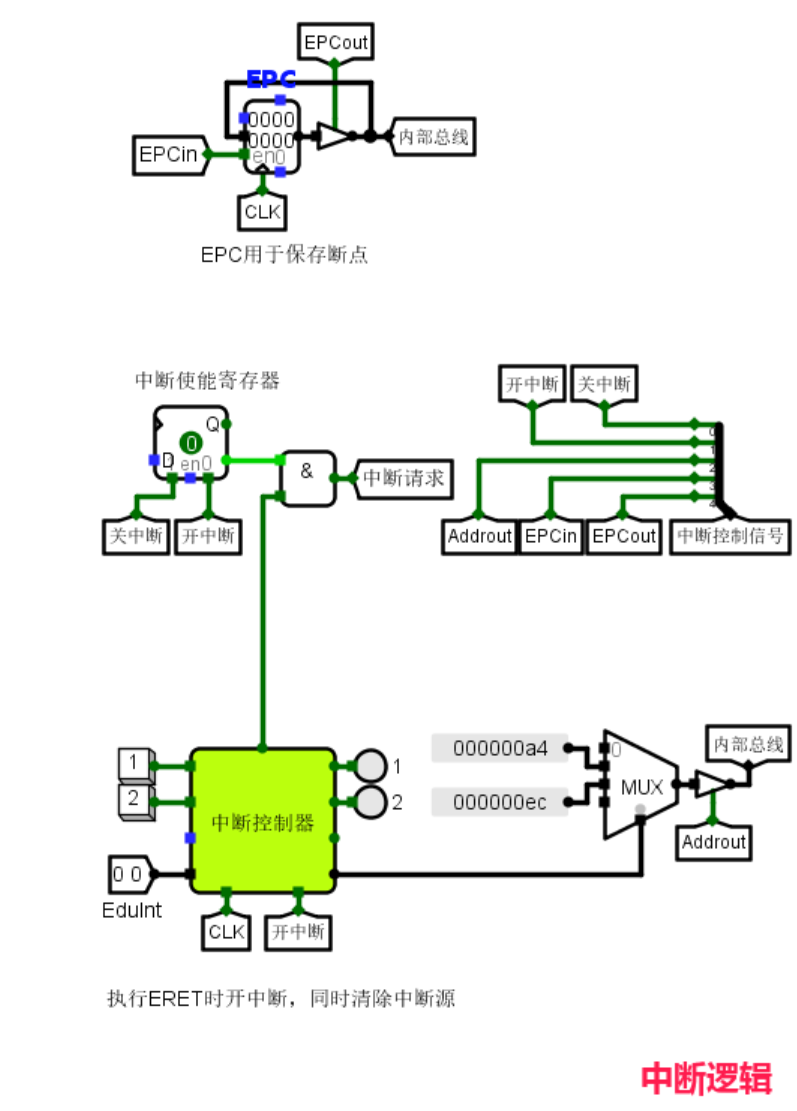


图1-11 中断逻辑

### 支持中断的现代时序硬布线控制器状态机设计

按状态图填写5号excel表，在IR为1时次态应为26即进入中断。对于状态15，应先判断equal是否为1，是则进入状态16，否则判断IR。填写完成的表格如图1-12所示，自动生成次态逻辑表达式后，即可在logisim中自动生成该电路。

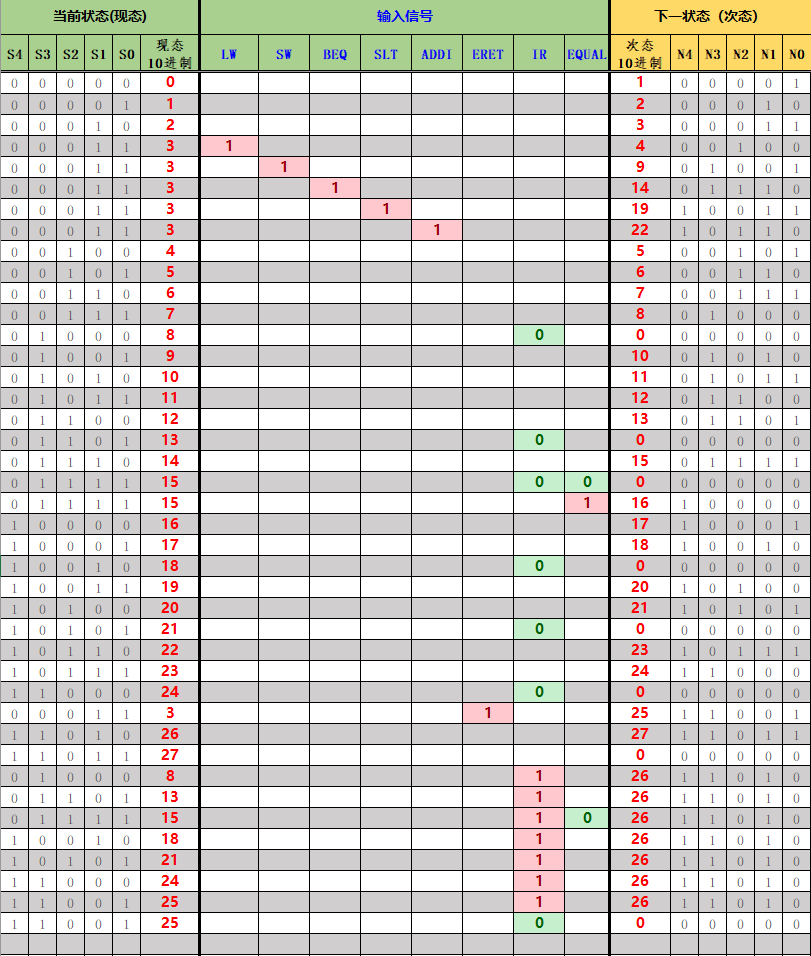


图1-12 状态机逻辑生成表

### 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，根据功能设计硬布线控制器。硬布线控制器状态机输出次态给状态寄存器，状态寄存器输出现态给状态机。

将1.2.4中所得的16进制微指令存入控制存储器。对于状态寄存器，应设置为下降沿有效。最终电路如图1-13所示。

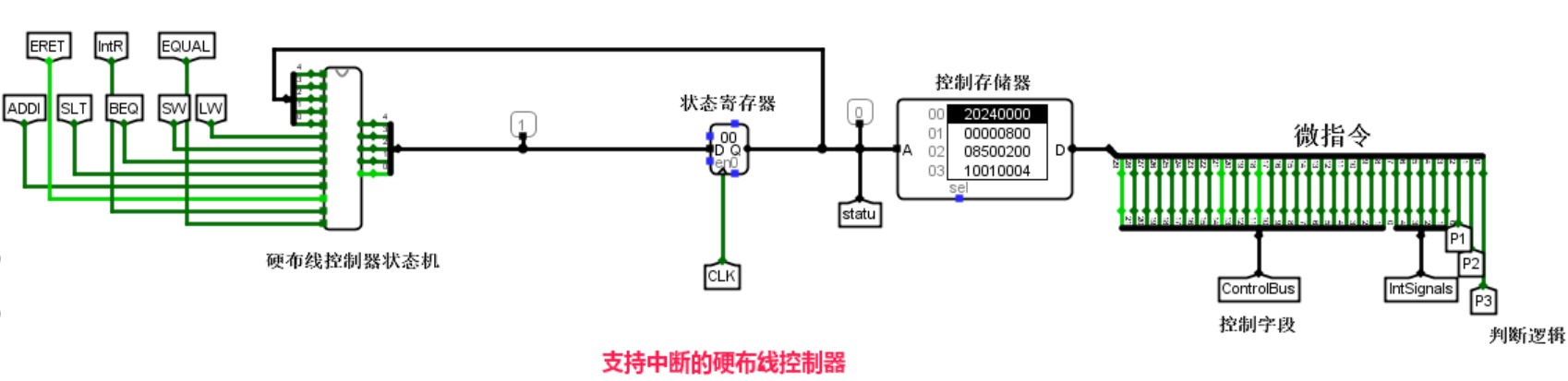


图1-13 支持中断的现代时序硬布线控制器状态机

## 实验步骤

1. 根据实验指引以及课程所学知识，设计上述解决方案，使用所提供的excel表格和circ文件，利用logisim工具实现所设计的解决方案。
2. 对照educoder所提供的测试用例及输出结果说明，对所设计并实现的电路进行测试与验证并作出相应调整。
3. 使用记事本打开设计好的circ文件，复制粘贴至educoder上进行测试，根据测试结果作出分析并进行进一步的调整与修改。

## 故障与调试

### 硬布线状态机问题

**故障现象：**硬布线状态机出现错误，无法通过educoder测试。

**原因分析：**在填写状态机逻辑生成表时，最初未注意到IR和EQUAL信号出现情况的不同组合，为在表格中填入0而只填入了1，导致在logisim中所生成的硬布线状态机出现问题，影响了测试。

**解决方案：**根据IR和EQUAL所出现的不同组合，按照状态图填入对应的0和1组合以及对应的次态，使状态机能够顺利按照预期工作。

## 测试与分析

根据实验要求，在带有中断功能的微程序单总线CPU的MEM中载入镜像文件sort-5-int.hex，使用ctrl+k开始运行，最终结果如图1-14所示。

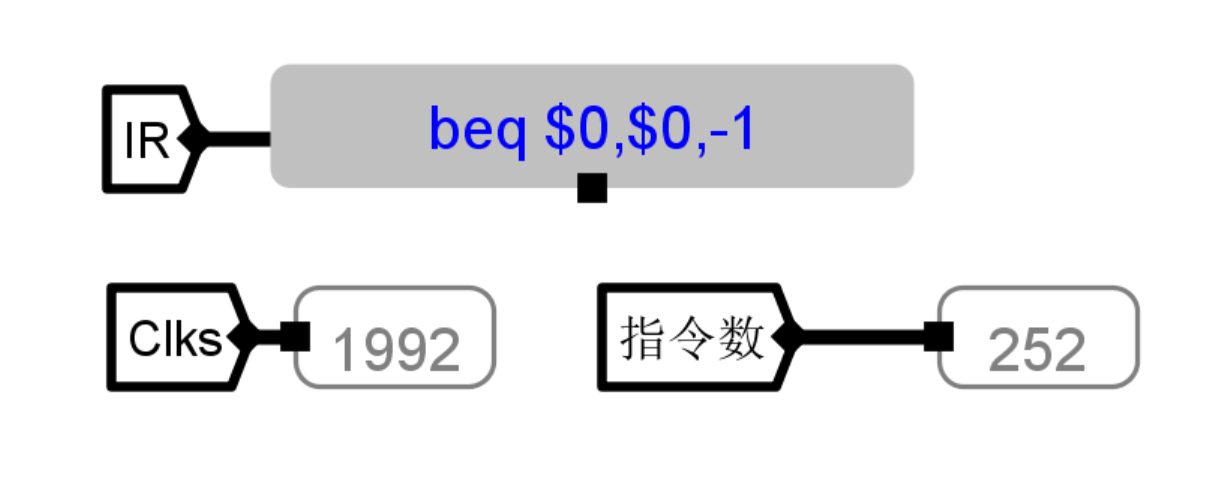


图1-14 最终运行结果

程序运行至1992节拍停下，指令计数为252，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。完成了有符号数的降序排序，MEM中排序结果如图1-15所示。

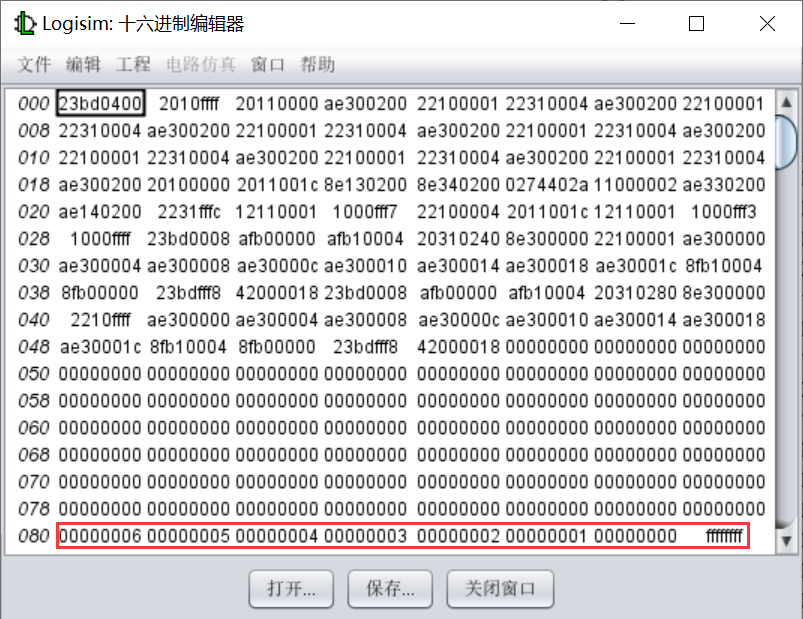


图1-15 内存MEM中有符号数降序排序结果

## 实验总结

本次实验主要完成了如下几点工作：

1. 设计了支持中断的微程序控制器，实现了支持中断的现代时序单总线结构的MIPS CPU。
2. 设计了现代时序的微程序指令，添加了中断处理的部分，实现了CPU中断功能的添加，完成了对标准测试程序sort-5-int.hex的调试和运行，达到了所预期的目标。

## 实验心得

1. 本次实验是MIPS现代时序中断机制实现，是在已经完成了现代时序的单总线CPU设计后的进阶，添加了中断的功能。通过本次的实验，我对CPU的组成和工作原理有了更加清晰的了解与认识，诸如CPU的各个内部部件的功能及其设计与实现、各个部件之间的相互联系与连接、CPU在这些部件的基础上运行的过程等等一系列知识。
2. 对于CPU的中断功能的实现也有了进一步的认识。对于处理单级中断的流程有了更深的认知，对于关中断、保存断点、执行中断以及开中断这一系列操作的具体流程有了更深刻的认知。
3. 进一步熟悉了使用logisim进行设计的操作，了解并使用了如译码器等各个部件，了解了使用logisim进行调试运行hex文件的操作。
4. 认识了Mars4\_5.jar工具的使用，了解了使用该工具对asm文件的操作。
5. 进一步熟悉了lw等数条指令的操作，深入理解了这些指令各个操作的具体在CPU中的流程。
6. 本次实验的指导清晰明确、简明扼要，能够让人快速上手实验操作，对于使用excel表格生成表达式和微指令也为实验带来了极大的便利，同时也能使学生在填写表格的同时能够进一步体会与熟悉相关的知识点。
7. 略微让人迷惑的是对于Mars4\_5.jar的使用以及对于sort-5-int.hex文件的调试运行的说明与指引较少，可以对这两方面稍微增加一些指导。

|  |
| --- |
| 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: 汪宇飞** |
| 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） | 课程目标1  工具应用  （10分） | 课程目标2  设计实现  （70分） | 课程目标3  验收与报告  （20分） | 最终评定  （100分） | | 得分 |  |  |  |  | |
| **指导教师签字:** |