***2025***



**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： |  |
| 学 号： |  |
| 姓 名： | [作者] |
| 电 话： |  |
| 邮 件： | [@qq.com](mailto:180544392@qq.com) |
| 完成日期： | 2025-06-30 |



# CPU设计实验

## 设计要求

利用logisim平台构建支持lw、sw、addi、slt、beq指令及单级中断的MIPS32单周期CPU。本实验需要依次实现MIPS指令译码器、微程序控制器、硬布线控制器、中断逻辑等模块。基于微程序控制器，将各模块组装成支持中断的微程序单总线CPU，并在实现的处理器上运行冒泡排序算法，及进行中断测试。

本实验中，指令译码器将32位MIPS指令字译码为对应的控制信号，为后续控制器提供基础判断。微程序控制器包含微程序入口查找逻辑、条件判别测试逻辑、微指令存储器等模块，通过微程序入口查找逻辑和条件判别测试逻辑，实现对不同指令和中断的流程控制。硬布线控制器采用状态机方式，根据当前状态、指令类型和条件反馈直接生成下一个状态和控制信号。中断逻辑包括中断请求信号检测、中断服务程序入口选择、EPC保存与恢复等，确保CPU能正确响应和返回中断。各功能部件之间通过总线互联，最终实现如图 1‑1所示的支持中断的微程序单总线CPU。

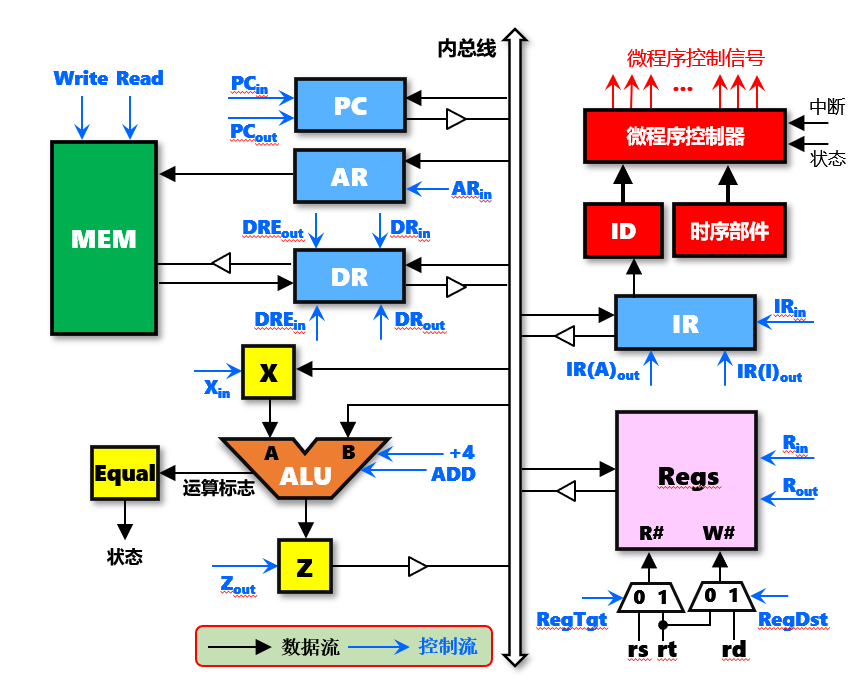


图 1‑1 支持中断的微程序单总线CPU

## 方案设计

### 指令译码器设计

**设计原理：**查阅MIPS手册，可以得到各指令的操作码如表1‑1 支持指令所示。

表1‑1 支持指令

|  |  |  |  |
| --- | --- | --- | --- |
| 助记符 | 格式 | op/funct | 功能描述 |
| lw | lw rt, offset(base) | 24H | 加载字 |
| sw | sh rt, offset(base) | 2BH | 存储字 |
| addi | addi rt,rs, immediate | 08H | 加立即数 |
| slt | slt rd, rs, rt | 00H/2AH | 小于置1 |
| beq | beq rs, rt, offset | 04H | 等于转移 |

其中，操作码op为32位指令字的31:26位，funct为指令字的5:0位。

**设计思路：**在logisim中可以通过分线器从32位指令字中提取出op和funct对应的字段，然后根据字段判断对应的指令。由于指令较少，因此指令译码器可借助比较器通过简单的组合逻辑完成，将比较的结果输出到对应的指令标志位，并设置额外的OtherInstr标志位用来表示非指定的5种指令之一。实现电路如图 1‑2所示。

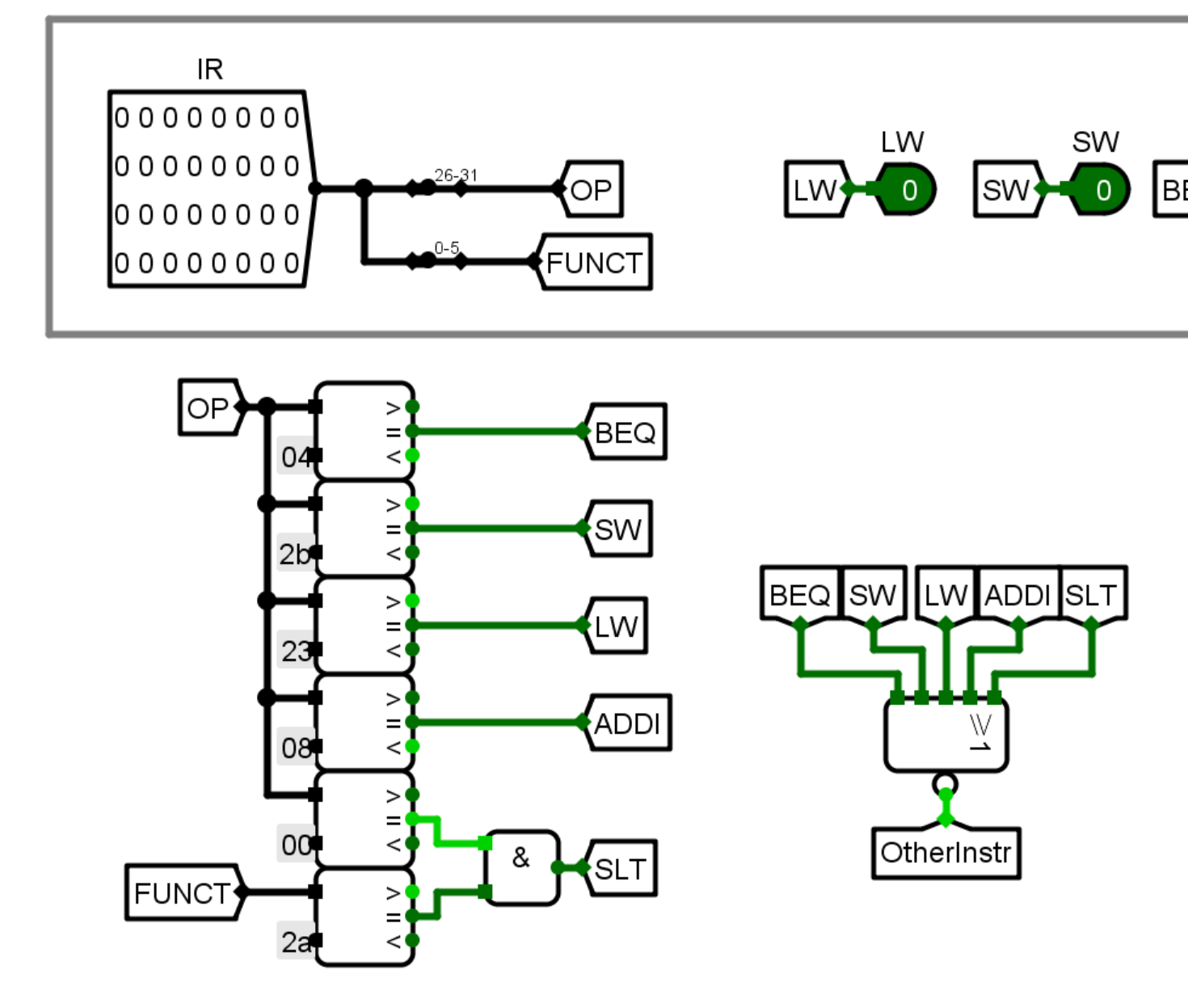


图 1‑2 指令译码器组合逻辑设计

### 微指令设计

**设计原理及思路：**理论上，微指令应包含控制信号字段、判别测试字段以及下址字段。为减少微指令长度，本实验中微指令设计为省略下址字段，由判别测试字段判断下址。因此本实验的微指令包含控制信号字段、判别测试字段两部分。

控制信号字段包含寄存器读写、ALU操作、存储器读写、PC更新等需要的信号，根据模型图1-1，包括PCout, DRout, Zout, Rout, IR(I)out, IR(A)out, DREout, PCin, ARin, DREin, DRin, Xin, Rin, IRin, PSWin, RegTgt, RegDst, Add, Add4, Slt, READ, WRITE,共22个信号；此外还需EPCout, EPCin, Addrout, STI, CLI共5个中断控制信号。

在设计判别测试字段前，需要分析地址转移情形：

1. 顺序转移，即PC自增1；
2. 指令入口转移，发生于取指周期后根据指令类型转移到对应入口地址；
3. beq分支跳转，发生于beq指令中满足相等条件；
4. 中断响应入口，发生于执行周期结束且发生中断；
5. 取指微程序入口转移，发生于执行周期结束且无中断发生，或中断结束。

将以上5种情形编号为0~4，即二进制(000)2~(100)2，使用S2S1S0表示。根据以上转移条件，设置P0、P1、P2三个判断测试位。加上ALU状态位Equal，各位的功能说明见表 1‑2 判断测试字段。

表 1‑2 判断测试字段

|  |  |  |  |
| --- | --- | --- | --- |
| 信号 | 输入输出 | 位宽 | 功能描述 |
| P0 | 输入 | 1 | 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 | 1 | 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 | 1 | 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 | 1 | 条件状态位，表示运算相等 |

综上所述，本实验设计微指令长度为30位，其中0~2为判断测试位，3~7为中断控制位，8~29为总线控制位。

### 微程序条件判别测试逻辑

**设计原理及思路：**根据判断测试字段3位P0、P1、P2，结合ALU状态位Equal以及中断请求信号IntR的不同排列组合，对应转移情形如表 1‑3 判别测试逻辑转移表所示。

表 1‑3 判别测试逻辑转移表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **输入 （填1或0，不填为无关项x）** | | | | | **输出 (只填写为1的情况)** | | |
| **P0** | **P1** | **P2** | **equal** | **IntR** | **S2** | **S1** | **S0** |
| **0** | **0** | **0** |  |  | **0** | **0** | **0** |
| **1** | **0** | **0** |  |  | **0** | **0** | **1** |
| **1** |  | **1** |  |  | **0** | **0** | **0** |
| **1** | **1** |  |  |  | **0** | **0** | **0** |
| **0** | **1** | **0** | **1** |  | **0** | **1** | **0** |
| **0** | **1** | **1** | **1** | **0** | **0** | **1** | **0** |
| **0** | **1** | **1** | **1** | **1** | **0** | **1** | **1** |
| **0** | **0** | **1** |  | **0** | **1** | **0** | **0** |
| **0** | **0** | **1** |  | **1** | **0** | **1** | **1** |
| **0** | **1** | **1** | **0** | **0** | **1** | **0** | **0** |
| **0** | **1** | **1** | **0** | **1** | **0** | **1** | **1** |

其中，空格子表示可为0或1任意一个。以上表格同时包含了冲突情形，例如，当同时发生beq分支跳转和中断时，将有限执行分支跳转。

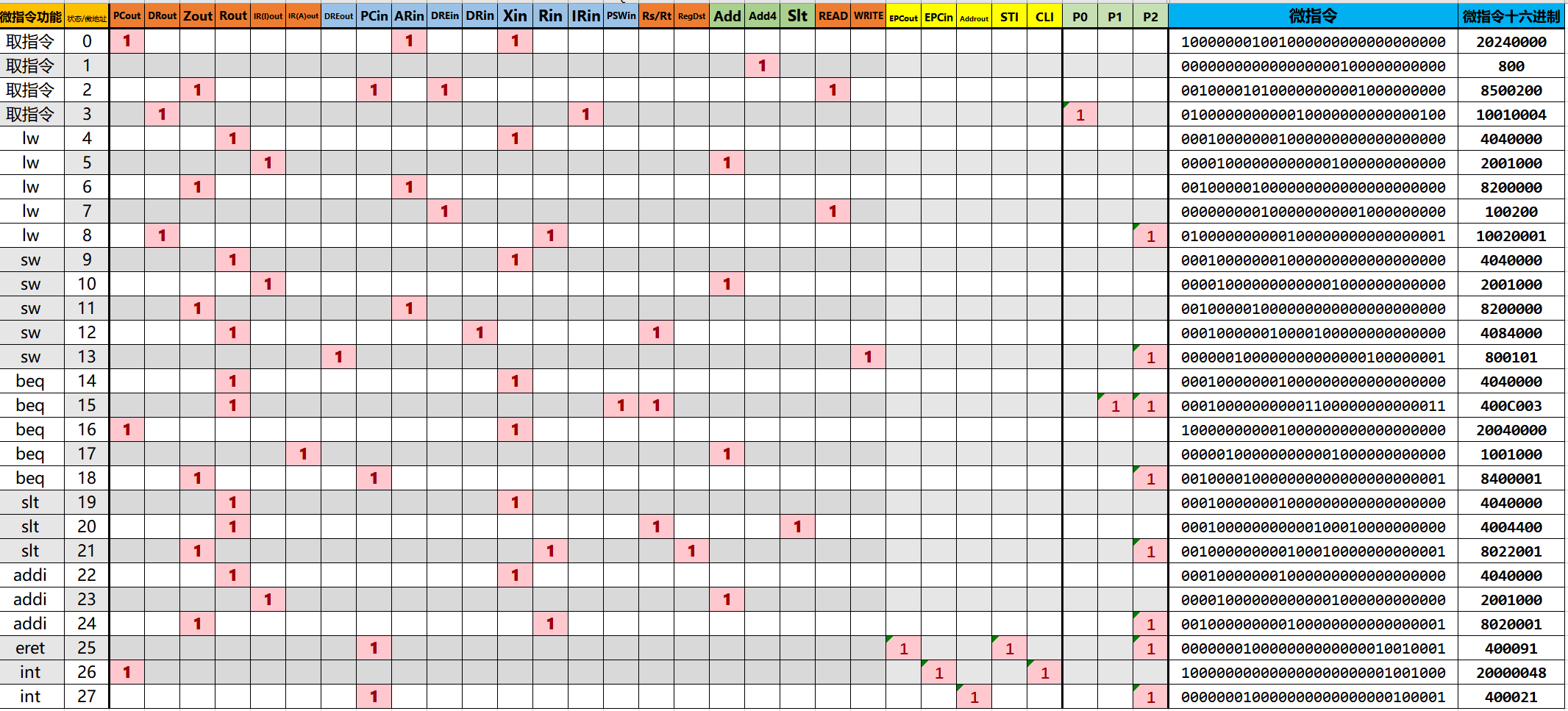
**设计实现：**根据以上表格可导出输出S2、S1、S0分别关于输入P0、P1、P2、Equal、IntR的逻辑表达式：

根据以上逻辑表达式，使用logisim的电路生成功能生成电路。

### 微程序及微程序入口查找逻辑设计

**设计原理及思路**：根据避免总线数据冲突的原则，设计取指周期及各指令执行周期的微程序如下表1-4。由该表可导出微指令程序并填入控制存储器，如表 1‑4。

表 1‑4 微程序设计



此时，还剩下的工作为入口查找逻辑实现。可列出微指令执行流程图如图 1‑3，其中S0-S27的编号可视为28行微指令的行号。那么，各个微程序入口分别为S0、S4、S9、S14、S19、S22、S25。其中，取指微程序入口可省略，固定为常数0。其余入口则需要根据输入的组合逻辑实现，列出入口表 1‑5。

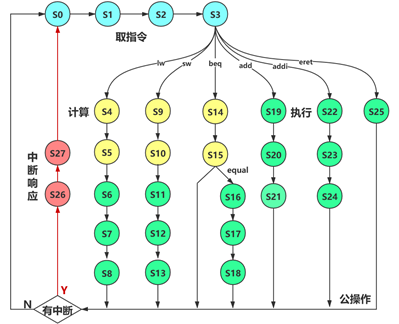


图 1‑3 微指令执行流程图

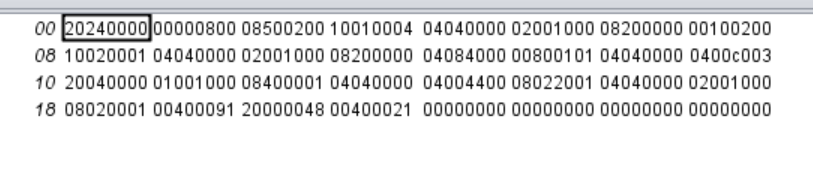


图 1‑4 控制存储器

表 1‑5 微程序入口查找表

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 |

由于共28行微程序，，故至少需要使用5个状态位表示行号，设为S4S3S2S1S0。

**设计实现：**根据表1-4可导出S4S3S2S1S0表达式，再根据logisim的电路生成功能生成入口查找逻辑电路如图 1‑5。

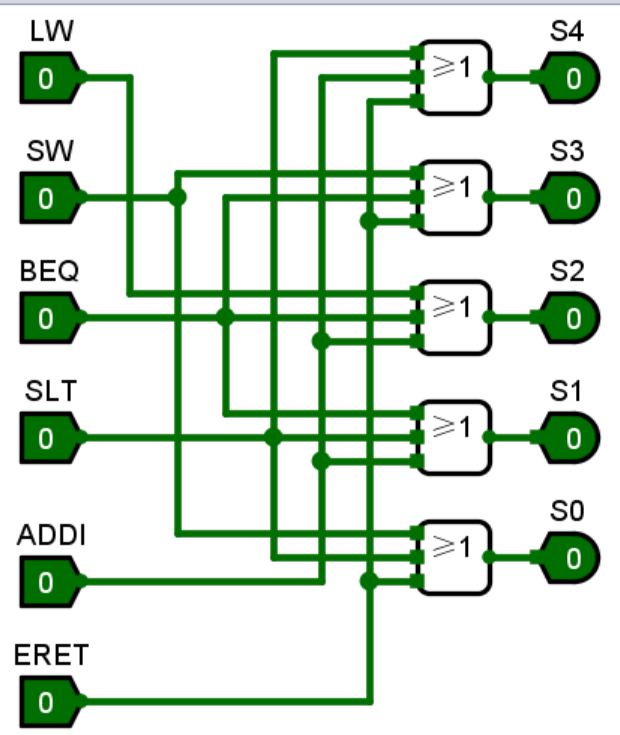


图 1‑5 入口查找逻辑电路

### 微程序控制器设计

**设计原理及思路：**将判断测试逻辑、微程序入口查找逻辑、控制存储器、微指令连接，得到微程序控制器，见下图 1‑6。

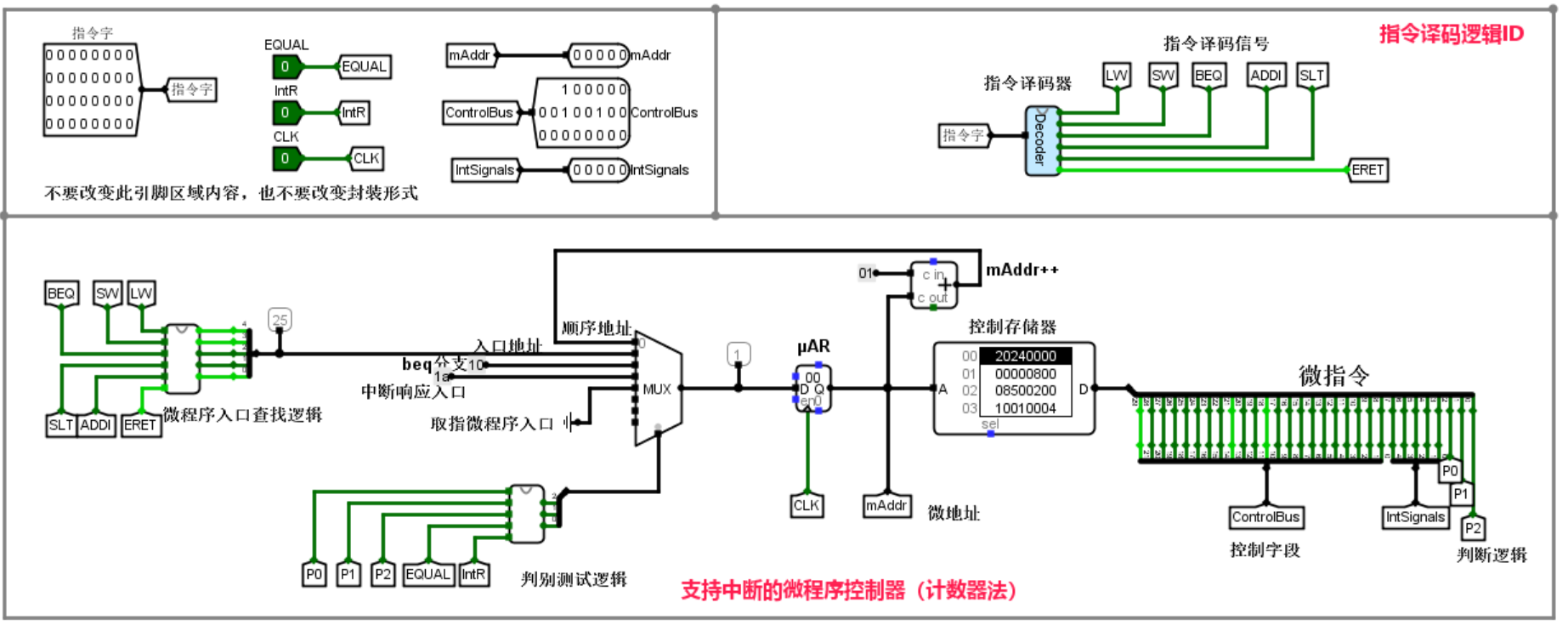


图 1‑6 微程序控制器实现

### 中断逻辑设计

**设计原理及思路：**根据给出的sort-5-int.asm代码，可以看到其中有两个中断服务程序，对应logisim中的中断控制器上的两个按钮。当点击按钮后，若处于开中断状态，则产生中断请求IntR，根据中断控制器的输出中断请求序号，选择相应地中断处理程序地址，当Addrout开时从CPU内部总线输出到到PC并执行。

**设计过程**：使用MARS(Mips Assemble and Run Simulator)程序找出程序中的中断处理程序地址，并配置到选择器。如图 1‑7，中断程序入口地址分别为0xa4和0xec。将两个地址配置到中断逻辑中如图 1‑8。

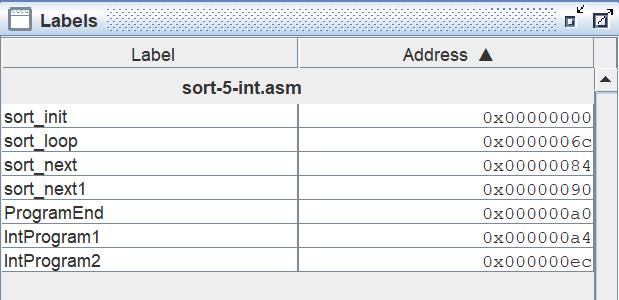


图 1‑7 中断处理程序各标签地址

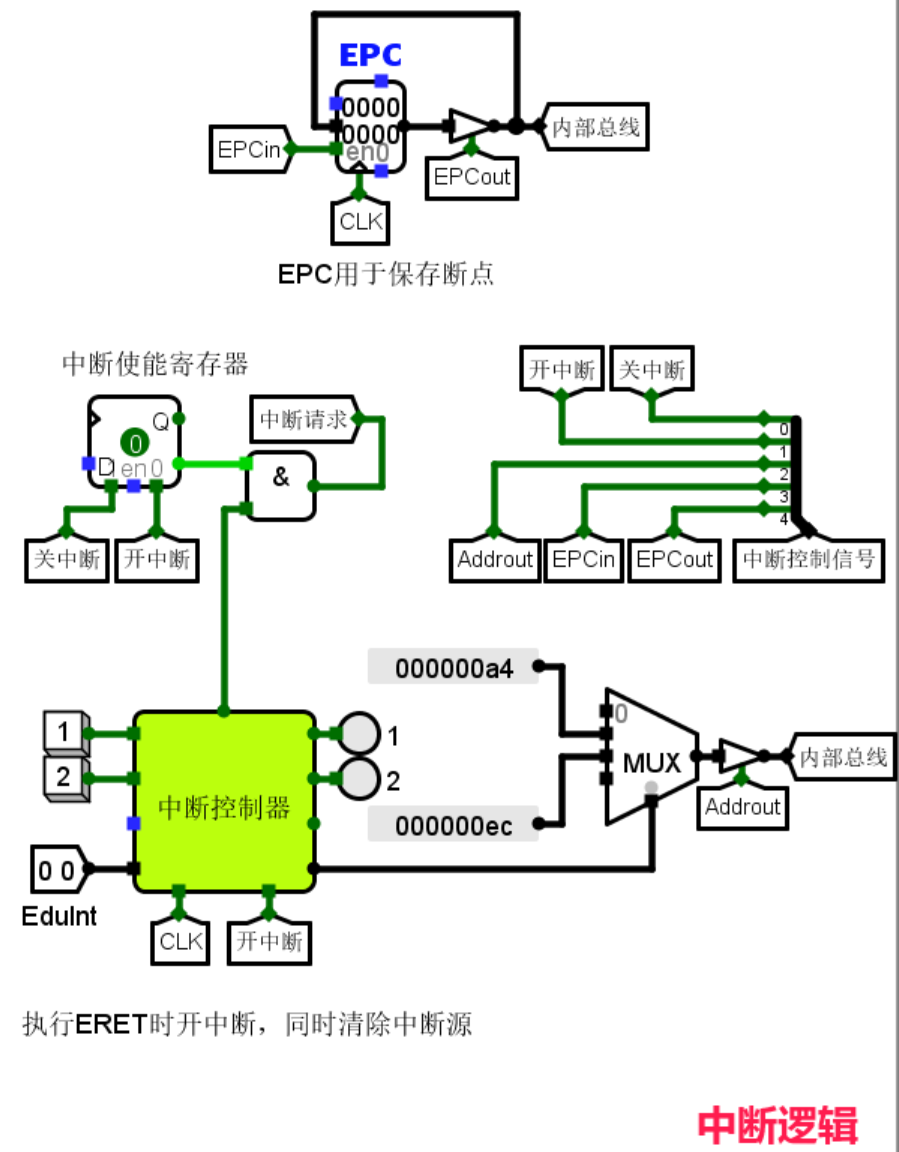


图 1‑8中断逻辑实现

### 支持中断的微程序单总线CPU设计

**设计原理及思路：**根据单总线CPU模型图 1‑1，在logisim中连接对应模块。实现如下图 1‑9。

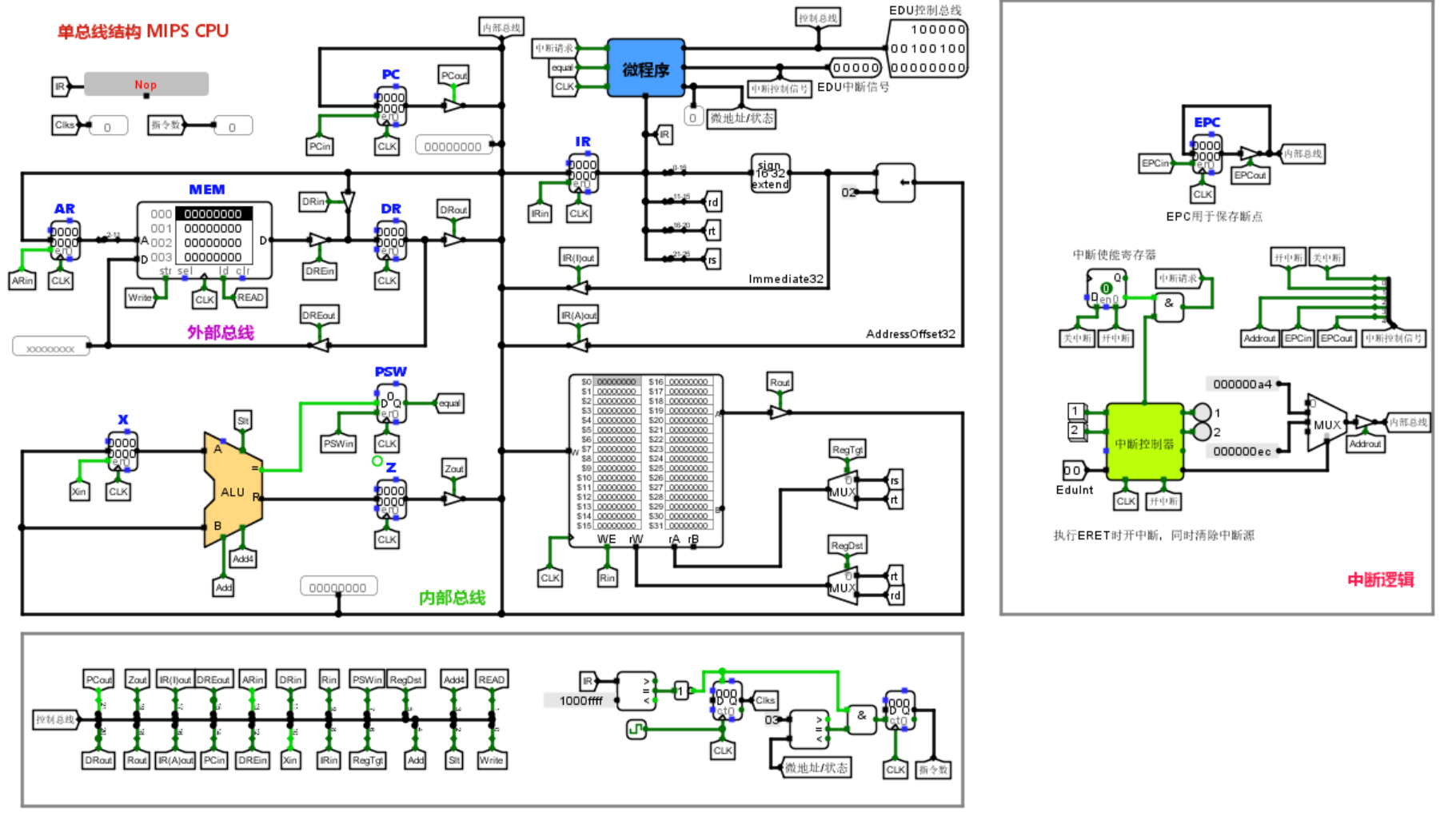
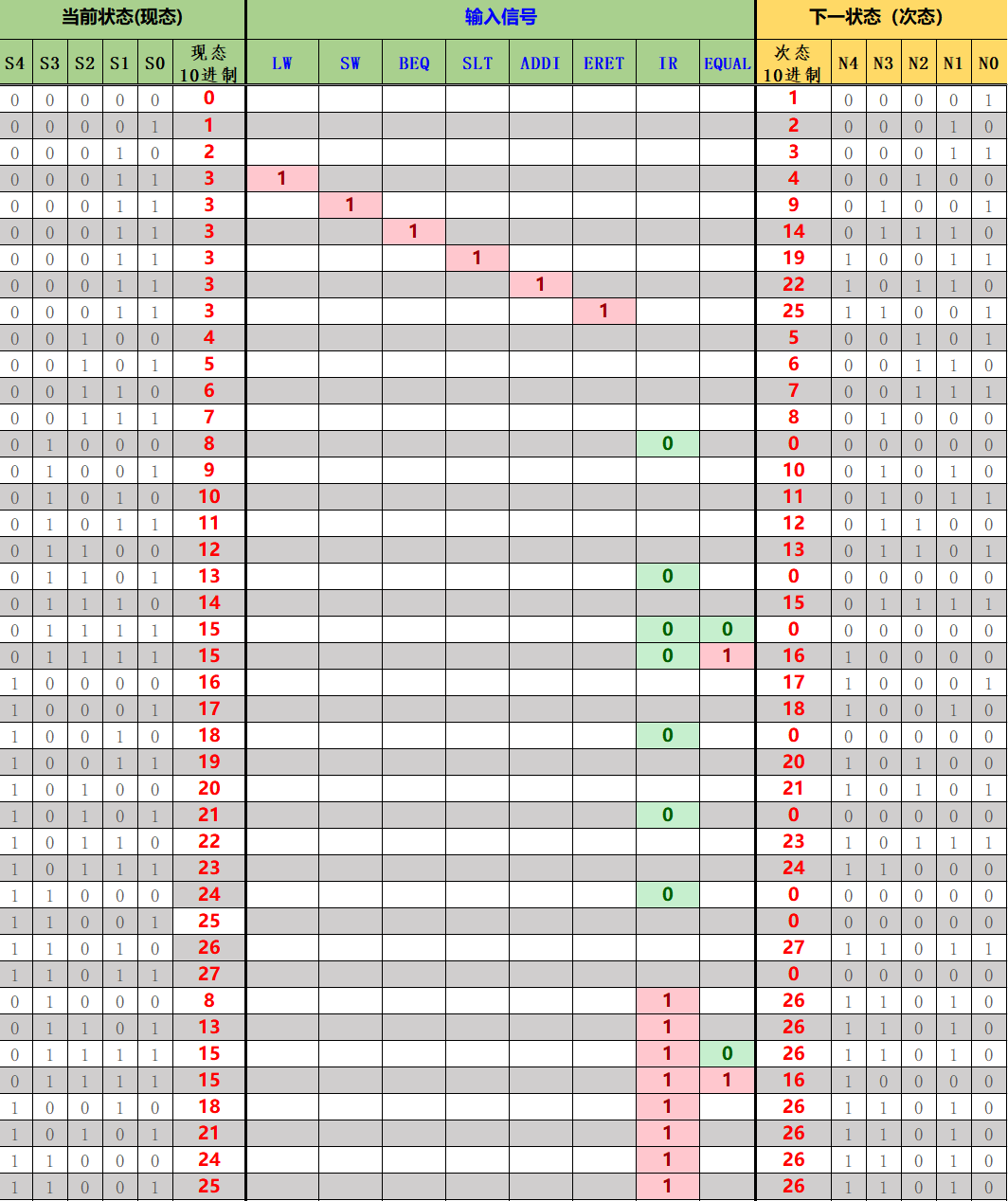


图 1‑9 支持中断的微程序单总线CPU

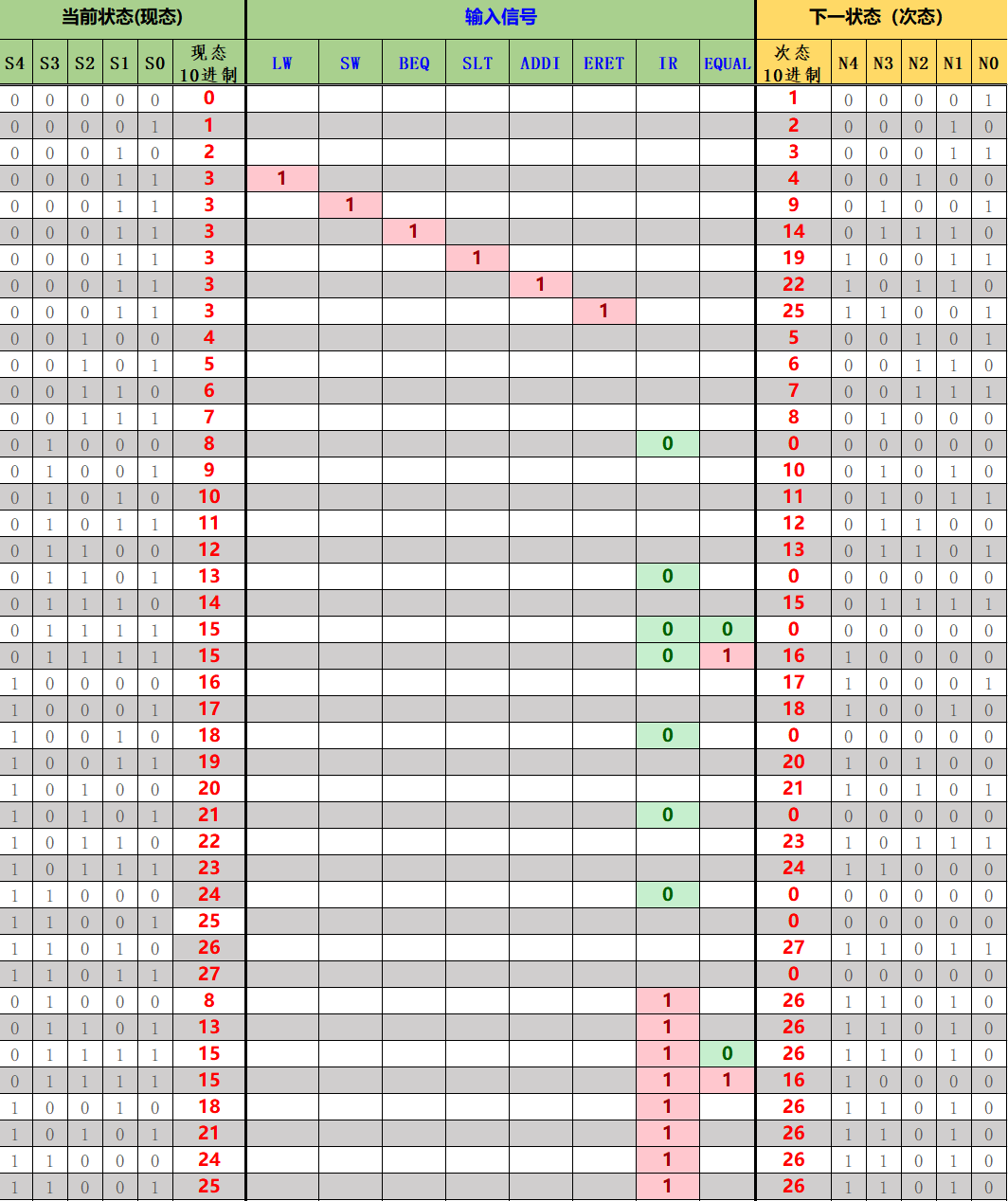
### 硬布线状态机及控制器设计

**设计原理、思路及过程：**硬布线状态机的输入不仅需要现态，还需要指令译码信号、Alu状态信号及中断请求信号。图1-3可视为状态转移图，列出状态转移表如下表 1‑6。

表 1‑6 硬布线控制器状态转移表



**. . . . . .**



其中，用N4N3N2N1N0表示次态的二进制编号。根据此表导出次态转移逻辑表达式，借助logisim的电路生成功能生成状态机转移电路。

根据实现的状态机，完成硬布线控制器如下图 1‑10。

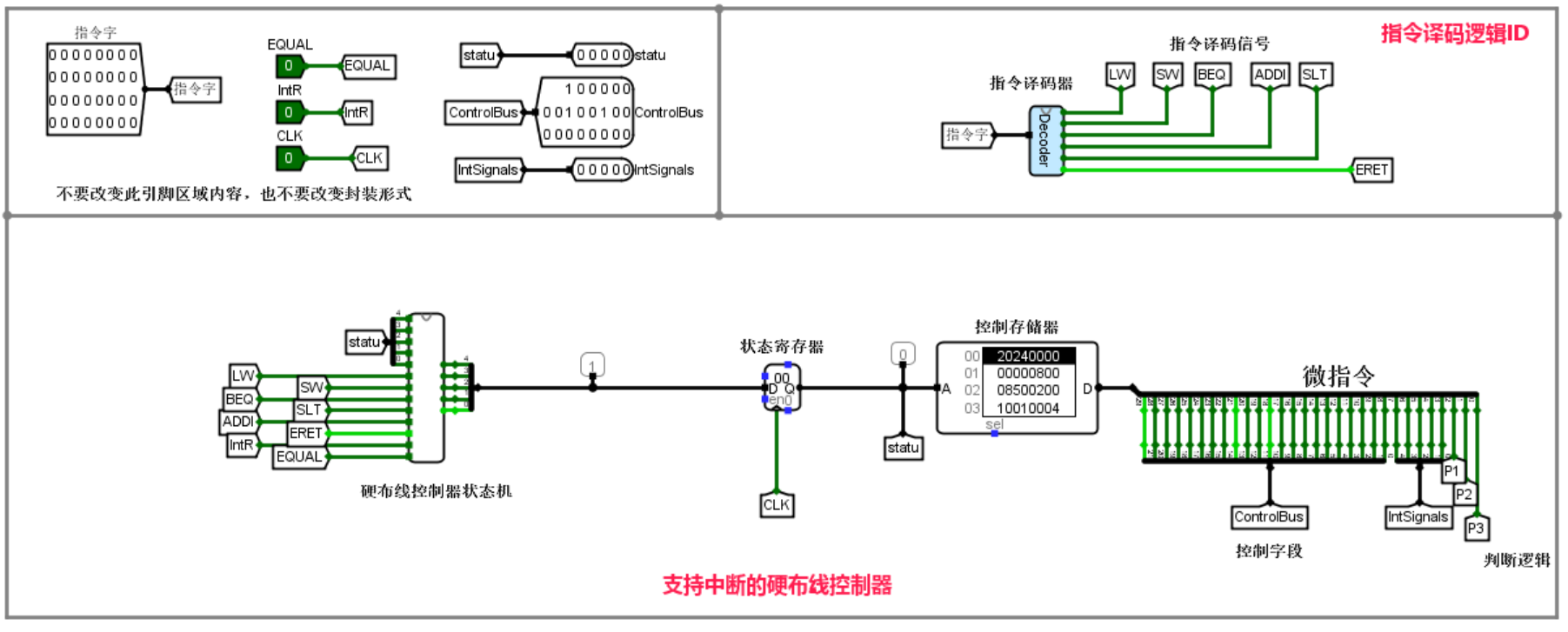


图 1‑10 硬布线控制器

## 实验步骤

1. 根据以上各模块设计原理及思路，完成各模块相应实现；
2. 各模块实现后，选择微程序控制器作为控制器，组装各模块形成完整的单总线CPU。
3. 向存储器模块中加载冒泡排序的16进制程序sort-5-int.hex，并开始模拟。
4. 当指令数不再增长后，在存储器模块上右键点击编辑内容，观察存储器中的内容。
5. 使用手指工具点击中断控制器旁边的按钮。
   1. 点击按钮1，观察存储器内容变化；
   2. 点击按钮2，观察存储器内容变化；
   3. 迅速点击按钮1和按钮2，观察存储器内容变化。
6. 停止模拟。

## 故障与调试

### 微程序条件判别测试逻辑错误

**故障现象：**头歌平台测试例未通过。

**原因分析：**如图 1‑11，错误的认为逻辑由上至下具有优先级。例如，当输入为01110时，输出不会是01x1x对应的010，而是输入01x1x和输入0x1x0的输出的逻辑或运算结果110。

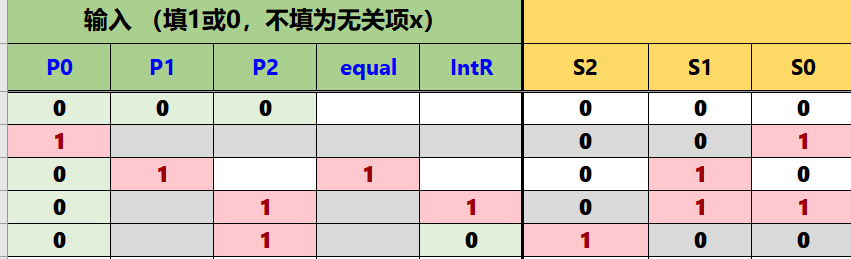


图 1‑11 判断测试错误逻辑

**解决方案：**补充输入的更多组合情形，如表 1‑3。

### 中断处理程序执行结果不符合描述

**故障现象：**在sort-5-int.asm程序中，给出的中断处理程序描述为：“中断服务程序1将字地址0x88开始的8个字数据累加1，中断服务程序2将字地址0x90开始的8个字数据累加2。”然而，如图 1‑12所示，实际执行效果为：中断服务程序1将字地址0x90开始的8个字数据累加1，中断服务程序2将字地址0xa0开始的8个字数据累减1。

**原因分析：**怀疑汇编代码和编译的16进制代码对应的程序不同。分析汇编代码，如图 1‑13，发现程序执行目的与编译代码的执行效果相同，故判断为备注描述错误。

**解决方案：**备注改为中断服务程序1将字地址0x90开始的8个字数据累加1，中断服务程序2将字地址0xa0开始的8个字数据累减1。

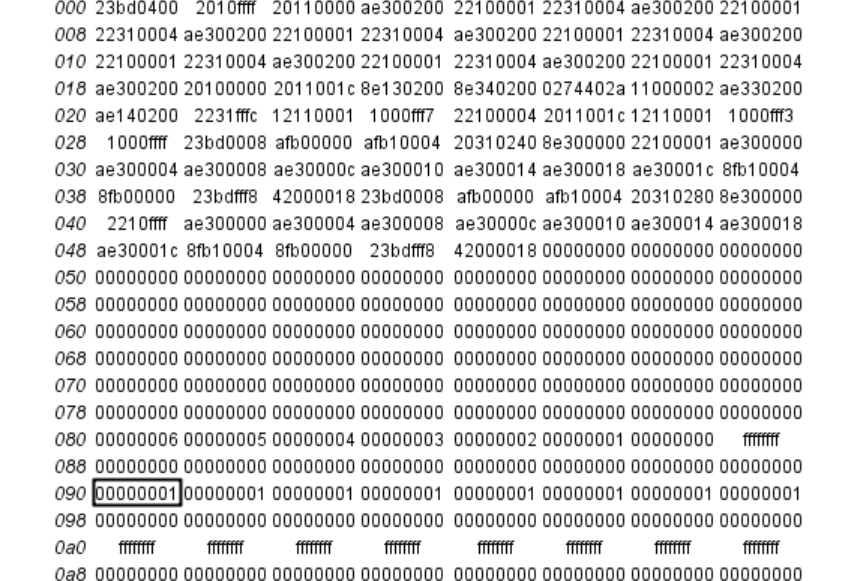


图 1‑12 中断程序执行结果

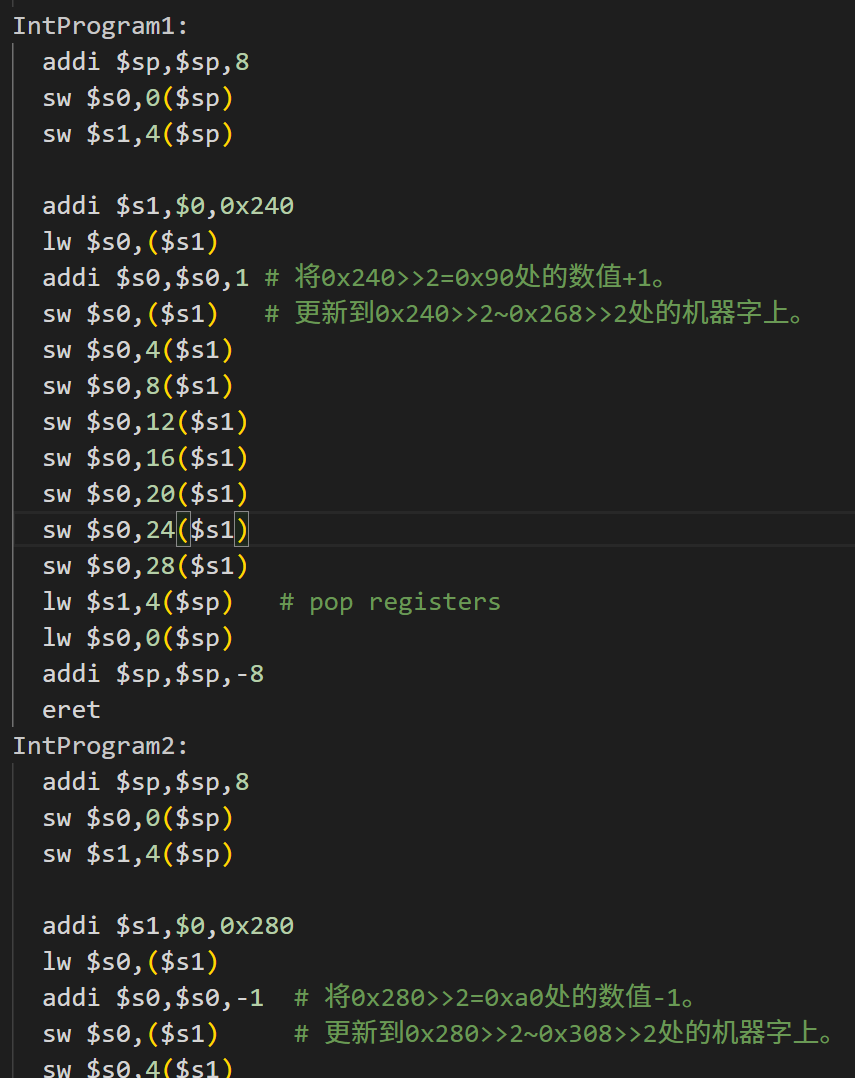


图 1‑13 中断程序的汇编代码

## 测试与分析

### 模块基础功能测试

如图 1‑14，各模块均通过头歌平台的所有测试例，说明各模块基础功能完善。



图 1‑14 头歌平台样例通过记录

### 冒泡排序执行测试

向存储器中手动加载sort-5-int.hex程序，按ctrl+k执行。根据asm汇编程序，执行过程中观察存储器中512>>2=0x80处连续8个机器字长的内存变化，如表 1‑7所示，可以明显的观察到冒泡排序的算法特征。

算法执行完成后，数字符合要求的降序排列。排序完成后，CPU无限执行beq $0,$0,-1。此时指令地址不再变化，指令数为252，如图 1‑15所示。

表 1‑7 冒泡排序执行过程

|  |  |
| --- | --- |
| Pass | 结果 |
| 0 |  |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |

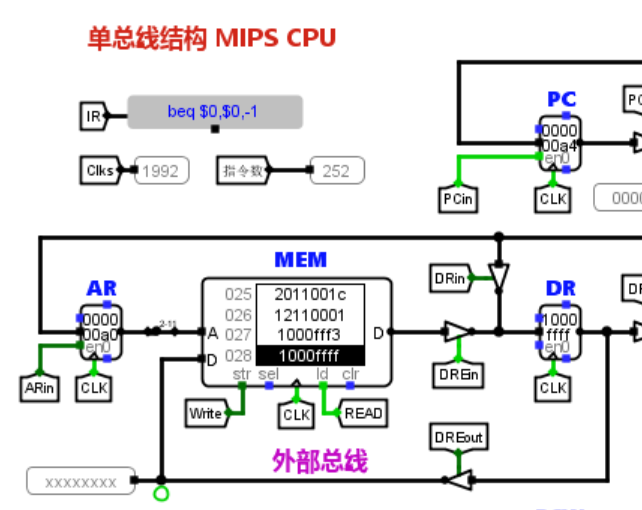


图 1‑15 冒泡排序执行结果

### 中断测试

1. 中断程序1功能测试

使用手指工具，点击中断控制器左侧按钮1，待指令数不再增长后，连续2次点击按钮1。执行结果如图 1‑16所示。说明中断程序1功能正常，且当短时间内发出多次重复的中断请求时，只执行1次，说明只支持单次中断，不支持嵌套中断，符合设计。





图 1‑16 中断程序1

1. 中断程序2功能测试

使用手指工具，点击中断控制器左侧按钮2，待指令数不再增长后，再次点击按钮2。如图 1‑17所示，功能正常。





图 1‑17 中断程序2

1. 中断优先级测试

暂停CPU运行，使用手指工具，依次点击中断控制器左侧按钮1和2。观察到先执行中断程序2，后执行中断程序1。执行结果如图 1‑18所示。



图 1‑18 依次点击按钮1和2

待指令数不再增长后，暂停CPU运行，使用手指工具依次点击按钮2和1。观察到先执行中断程序2，后执行中断程序1。执行结果如图 1‑19所示。



图 1‑19 依次点击按钮2和1

测试结果说明，中断程序2的优先级高于中断程序1，符合中断控制器功能。

## 实验总结

本次实验主要完成了如下几点工作：

1. 完成方案总结：本实验中设计了微指令格式及内容，完成了lw、sw、beq、slt、addi、eret指令的微程序设计，依次实现了指令译码器、微程序入口查找逻辑、条件判别测试组合逻辑、硬布线状态机等模块，最终搭建了支持中断的微程序单总线CPU，完成了数据通路和控制通路的组装与调试。
2. 功能总结：本次实验成功在实现的微程序单总线CPU上加载并运行冒泡排序程序，集成了中断处理机制，实现了中断请求检测、响应和返回。

## 实验心得

1. 本次实验最大的收获是深入了解了单周期CPU的结构与工作原理，通过亲手设计和搭建MIPS单周期CPU，让我更加熟悉了Logisim工具的使用。但是总感觉不过瘾，一是虚拟平台给人的感受不甚强烈，二是闯关模式将框架给出太多，三是线下课太少，上的没有感觉。
2. 实验建议：希望可以完善头歌平台的关卡说明，以及修改关卡顺序，比如先设计微程序，再设计条件判断测试逻辑，再设计入口查找逻辑，正如本报告中的顺序，这样的顺序更符合控制器模型建立的过程。
3. 实验感想：写报告时，龙芯3C6000发布，创造了全新的指令集架构，实现了指令集上的自主，更实现了硬件加速ABI翻译，大大降低了指令集模拟的性能损失。这一切让刚学完组原和汇编的我深受触动，既感到亢奋，又感到肩头沉甸甸的责任。感谢课程组在课程实验上做出的改革与付出，感谢老师一学期以来的教导。我将带着课程的收获，继续探索计算机系统的奥秘，期待国产芯片在教材和现实中绽放更多光彩。

|  |
| --- |
| 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:** [作者] |
| 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:** |