***2023***



**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： | CS21XX |
| 学 号： | U202111224 |
| 姓 名： | 郭德纲 |
| 电 话： | 1345565666 |
| 邮 件： | [13456@qq.com](mailto:13456@qq.com) |
| 完成日期： | 2023-05-30 |



# CPU设计实验

**只写CPU设计中“MIPS现代时序中断机制实现”，总篇幅16页以内。**

**请仔细阅读所有的批注，阅读理解后删除批注**

**模板各个标题下面的内容仅是举例，作者应依照自己思想重写该部分内容**

**请仔细阅读所有的批注，阅读理解后删除批注**

**模板各个标题下面的内容仅是举例，作者应依照自己思想重写该部分内容**

## 设计要求

使用两种不同的方案实现MIPS现代时序中断机制，两种不同的方案分别是微程序控制器和硬布线控制器。这两种控制器在输入和输出引脚上完全相同，功能也完全相同，仅内部实现方式有所差别。

微程序控制器的基本思想是仿照程序设计的基本方法，将所有指令的控制信号按照一定的跪着编码成微指令，将所有的微指令存放在一个只读存储器中，即控制存储器（控存），通过按照一定顺序执行控存中的微指令，就可以产生一定顺序的控制信号，从而实现一定的程序功能。

硬布线控制器采用组合逻辑电路的设计思想，将控存中微指令的地址看作一定的状态，通过分析程序指令执行的顺序得到程序的状态转移图，利用该状态转移图实现状态控制机，从而实现状态转移（微指令地址转移）的功能。

表1-1描述了这两种状态机的引脚与对应的功能。

表1-1 状态机引脚与功能描述

|  |  |  |  |
| --- | --- | --- | --- |
| 引脚 | 输入/输出 | 位宽 | 功能描述 |
| IR | 输入 | 32 | MIPS指令字 |
| equal | 输入 | 1 | 条件反馈信号，表示运算相等 |
| IntR | 输入 | 1 | 中断请求信号 |
| mAddr/Statu | 输出 | 5 | 当前微指令的地址/程序的状态 |
| ControlBus | 输出 | 22 | 微指令控制字段 |

其中微指令的组成如图1-1所示。IntSignals表示中断信号。P0~P2用于判断所要执行的分支的判别测试位。ControlBus表示控制总线。

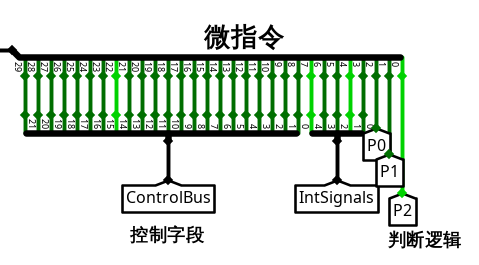


图1-1 IR组成图

图1-2描述了MIPS数据通路中指令执行的过程。

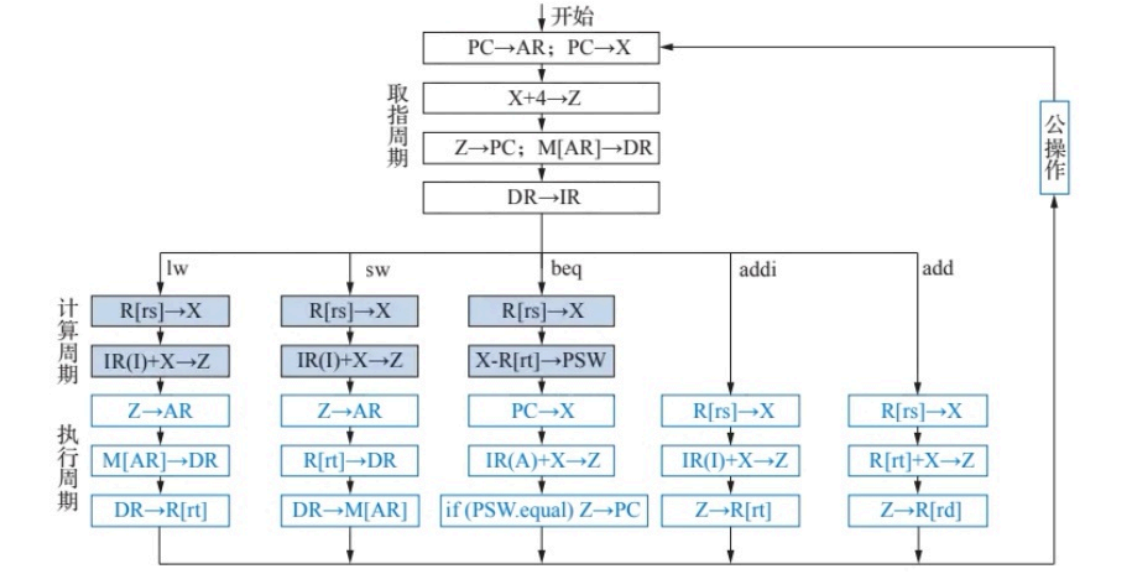


图1-2 指令执行过程图

这里的“公操作”在本实验中就是中断信号的判断以及响应。

图1-2描述了图1-3对应的状态转移图。

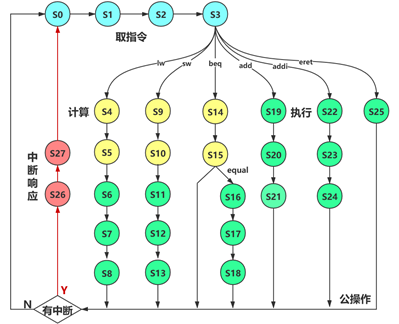


图1-3 状态转移图

其中，每一个状态对应上文所说的一个微指令地址/程序的状态，也即表1-1中描述的mAddr/Statu引脚。

本实验的要求是使用硬件实现上述转移逻辑，最终实现MIPS现代时序中断机制。

## 方案设计

本实验在设计的过程中采用分模块设计的方法。本实验需要设计实现模块有MIPS指令译码器、支持中断的微程序入口查找逻辑、支持中断的微程序条件判别测试逻辑、支持中断的微程序控制器、支持中断的微程序单总线CPU、支持中断的现代时序硬布线控制器状态机以及支持中断的现代时序硬布线控制器。

### MIPS指令译码器

所有方案应将设计思路和设计原理、过程写清楚，为什么这样设计，各部件之间的关系，仅仅粘贴一张电路图是不合格的报告。

MIPS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。

表1-2描述了MIPS指令译码器的引脚与对应的功能。

表1-2 MIPS指令译码器引脚与功能描述

|  |  |  |  |
| --- | --- | --- | --- |
| 引脚 | 输入/输出 | 位宽 | 功能描述 |
| IR | 输入 | 32 | MIPS指令字 |
| LW | 输出 | 1 | 当前指令为MIPS的LW指令 |
| SW | 输出 | 1 | 当前指令为MIPS的SW指令 |
| BEQ | 输出 | 1 | 当前指令为MIPS的BEQ指令 |
| ADDI | 输出 | 1 | 当前指令为MIPS的ADDI指令 |
| SLT | 输出 | 1 | 当前指令为MIPS的SLT指令 |
| OtherInstr  (ERET) | 输出 | 1 | 当前指令为MIPS的其他指令，在本实验中，该引脚用于表示中断返回ERET信号 |

在实现的过程中，我们需要根据MIPS指令手册将IR的操作码OP以及功能码FUNC提取出来，再根据手册上各个指令OP以及FUNC的值，使用组合逻辑电路将这两个值转换成各个指令的译码信号。

具体实现见：

### 支持中断的微程序入口查找逻辑

根据MIPS指令译码器输出的译码信号获取微程序的入口地址/状态，即图1-4中的部分地址。



图1-4 微程序入口地址图

表1-3描述了支持中断的微程序入口查找逻辑的引脚与对应的功能。

表1-3支持中断的微程序入口查找逻辑引脚与功能描述

|  |  |  |  |
| --- | --- | --- | --- |
| 引脚 | 输入/输出 | 位宽 | 功能描述 |
| LW | 输入 | 1 | 当前指令LW指令 |
| SW | 输入 | 1 | 当前指令SW指令 |
| BEQ | 输入 | 1 | 当前指令BEQ指令 |
| ADDI | 输入 | 1 | 当前指令ADDI指令 |
| SLT | 输入 | 1 | 当前指令SLT指令 |
| ERET | 输入 | 1 | 当前指令ERET指令 |
| S4-S0 | 输出 | 各1位 | 微程序入口地址的二进制编码 |

在实现的过程中，我们需要用组合逻辑电路将各个信号的输入转化成微程序入口地址的二进制编码形式（S4~S0）。

具体实现见：

### 支持中断的微程序条件判别测试逻辑

支持中断的微程序条件判别测试逻辑完成在程序遇到分支时对程序的下一条微指令的地址做出选择的功能。本实验中包含的分支主要包括判断相等时的分支以及判断中断请求到来时的分支。

表1-4描述了条件判别测试逻辑的引脚与对应的功能。

表1-4条件判别测试逻辑引脚与功能描述

|  |  |  |  |
| --- | --- | --- | --- |
| 引脚 | 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 |
| IntR | 输入 | 1 | 条件状态位，中断请求信号 |
| S2-S0 | 输出 | 各1位 | 选择分支地址时的多路选择器控制信号 |

其中用于多路选择器的控制信号S2~S0在十进制下的取值分别表示不同的分支序号。分支序号对应的分支如表1-5所示。

表1-5 S2~S0的取值对应的分支

|  |  |
| --- | --- |
| S2~S0的十进制表示 | 分支地址 |
| 0 | 顺序地址（控存中下一条指令的地址） |
| 1 | 各个指令的入口地址 |
| 2 | BEQ分支地址 |
| 3 | 中断响应入口地址 |
| 4 | 取值微程序入口地址 |

在实现的过程中，我们需要将测试判别位（P0～P2）以及条件判别位（equal、IntR）通过组合逻辑电路转化成多路选择器的控制信号（S2~S0），并使得多路选择器的选择跟和表1-5一致。

具体实现见：

### 支持中断的微程序控制器

该部分利用1.2.1～1.2.3中实现的各个部件进行拼接组装，实现微程序控制器。并在控制存储器（控存）中输入各个微地址相应的微指令编码。

表1-6描述了微程序控制器的引脚与对应的功能。

表1-6 微程序控制器的引脚与功能描述

|  |  |  |  |
| --- | --- | --- | --- |
| 引脚 | 输入/输出 | 位宽 | 功能描述 |
| IR | 输入 | 32 | MIPS指令字 |
| equal | 输入 | 1 | 表示运算相等 |
| IntR | 输入 | 1 | 中断请求信号 |
| mAddr | 输出 | 5 | 当前微指令地址 |
| ControlBus | 输出 | 22 | 微指令控制字段 |

在实现的过程中，我们首先将IR输入到1.2.1中设计的指令译码器中，得到指令译码之后产生的控制信号（LW、SW、BEQ、ADDI、SLT以及ERET），接着将控制信号输入到1.2.2中设计的微程序入口查找逻辑电路中，获取各个信号对应的入口地址。再将程序的顺序地址、各个信号对应的入口地址、BEQ分支地址、中断响应入口地址以及取指指令的入口地址依次连接到多路选择器的各个选择引脚处，并使用判别测试逻辑生成的多路选择器控制信号来控制多路选择器。最后将多路选择器输出的微指令地址输入到微地址寄存器（uAR）中，再将uAR的输出端口连接到控制存储器中，作为读取微指令的地址。

具体实现见：

### 支持中断的微程序单总线CPU

使用给定的中断控制器、中断使能寄存器以及中断断点寄存器（EPC）设计中断逻辑，使得微程序单总线CPU支持中断。在实现的过程中首先利用中断控制器以及中断使能寄存器生成中断请求信号（IntR），接着利用中断控制器提供的中断号（IntNo）通过多路选择器选择相应的中断响应程序入口地址，再将中断响应程序的入口地址通过内部总线输入到程序计数器PC中。

具体实现见：

### 支持中断的现代时序硬布线控制器状态机

硬布线控制器状态机是硬布线控制器的核心部件，其作用是使用组合逻辑电路实现状态的转移，在本实验中，每一个状态对应控制寄存器中的一个微指令的微地址。本实验中实现的状态转移为图1-3中给出的状态转移图。

表1-7描述了硬布线控制器状态机的引脚与对应的功能。

表1-7硬布线控制器状态机的引脚与功能描述

|  |  |  |  |
| --- | --- | --- | --- |
| 引脚 | 输入/输出 | 位宽 | 功能描述 |
| S4-S0 | 输出 | 各1位 | 现态的二进制编码 |
| LW | 输入 | 1 | 当前指令LW指令 |
| SW | 输入 | 1 | 当前指令SW指令 |
| BEQ | 输入 | 1 | 当前指令BEQ指令 |
| ADDI | 输入 | 1 | 当前指令ADDI指令 |
| SLT | 输入 | 1 | 当前指令SLT指令 |
| ERET | 输入 | 1 | 当前指令ERET指令 |
| equal | 输入 | 1 | 表示运算相等 |
| IntR | 输入 | 1 | 中断请求信号 |
| N4~N0 | 输出 | 各1位 | 次态的二进制编码 |

利用组合逻辑电路实现现态到次态的状态转移，达到与图1-3中给出的状态转移过程一样的效果。

具体实现见：

### 支持中断的现代时序硬布线控制器

利用1.2.6中设计的硬布线控制器状态机实现硬布线控制器。硬布线控制器的引脚及其实现的功能与微程序控制器相同，在此不再赘述。不一样的地方在于，硬布线控制器使用状态机微指令地址的选择，根据输入的控制信号与当前状态选择相应的次态。

具体实现见：

## 实验步骤

### 实现支持中断的MIPS指令译码器

1. 使用MIPS指令手册，查看各个指令对应的操作码。如下表1-8所示。

表1-8各个指令对应的操作码

|  |  |  |  |
| --- | --- | --- | --- |
| 指令 | OP二进制 | OP十六进制 | 备注 |
| LW | 100011 | 0x23 |  |
| SW | 101011 | 0x2B |  |
| BEQ | 000100 | 0x04 |  |
| ADDI | 001000 | 0x08 |  |
| SLT | 000000 | 0x00 | FUNC=0x2A |

1. 分别使用位宽为6的比较器将输入的OP与各指令操作码进行比较，若相等则输出相应的控制信号。注意到SLT指令还要求功能码FUNC为0x2A，在判断SLT指令时还需要使用与门将输入FUNC与0x2A比较的结果与OP的比较结果一起进行判断。在Logisim中连线如下图1-5所示。

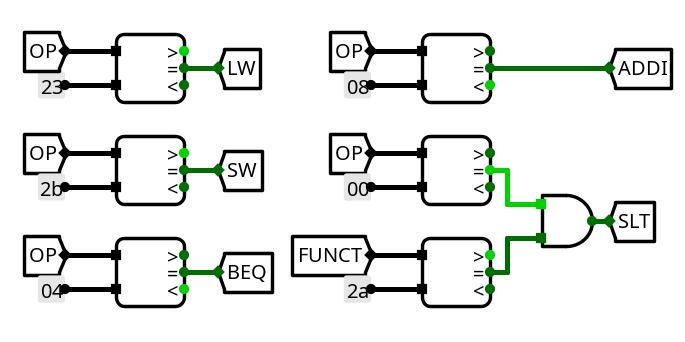


图1-5 指令译码

1. 对于本实验中的中断返回指令，在指令译码器中设计为其他指令，使用OrtherInstr作为输出信号。当2）中的指令信号全部为0时，OrtherInstr为1，因此使用非门进行判断。实现如下图1-6所示。

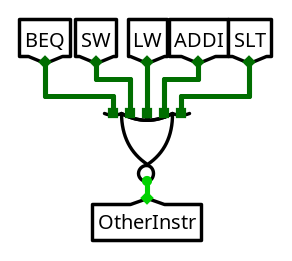


图1-6 其他指令的判断

### 实现支持中断的微程序入口查找逻辑

1. 根据图1-4可以得出，机器指令译码信号对应的微程序入口地址如下表1-9所示。

表1-9各个信号对应的微程序入口地址

|  |  |  |
| --- | --- | --- |
| 指令 | 入口地址10进制 | S4~S0 |
| LW | 4 | 00100 |
| SW | 9 | 01001 |
| BEQ | 14 | 01110 |
| SLT | 19 | 10011 |
| ADDI | 22 | 10110 |
| ERET | 25 | 11001 |

1. 根据上表1-9填写6号EXCEL表格，自动生成译码信号与微程序入口地址之间的逻辑关系。如下图1-7所示。

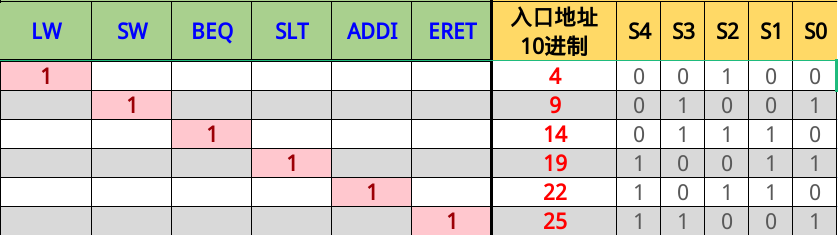


图1-7 译码信号对应的入口地址

1. 利用EXCEL自动生成的逻辑代数关系式在Logisim中自动生成电路图。生成的电路图如下图1-8所示。

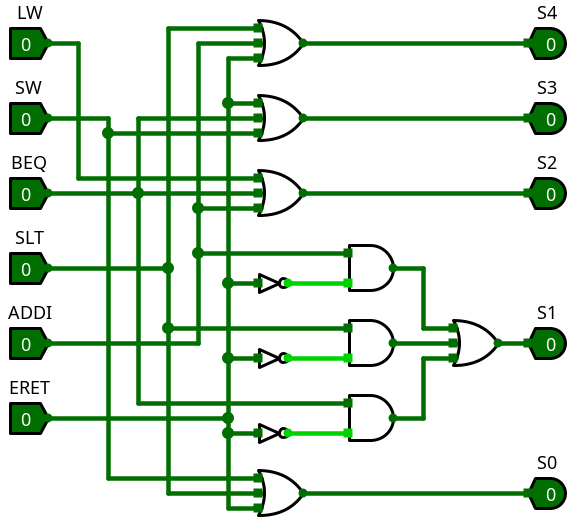


图1-8 指令入口地址查找逻辑

### 实现支持中断的微程序条件判别测试逻辑

### 根据测试判别位的输入以及状态条件位的输入输出多路选择器的控制信号。

### 当测试判别位全为0时，表示程序的状态并不处在任何一个要进行跳转的状态，因此输出的多路选择器控制信号应为0，表示顺序地址。

### 当测试判别位的P0为1时，表示当前程序需要根据控制信号进行地址跳转，因此输出的多路选择器控制信号应为1，表示指令入口地址。

### 当测试判别位的P1为1且equal信号为1时，表示需要跳转到BEQ分支地址，因此输出的多路选择器控制信号应为2，表示BEQ分支地址。

### 当测试判别位的P2为1且IntR为1时，表示要跳转到中断处理程序的入口地址，因此输出的多路选择器控制信号应为3，表示中断处理程序入口地址。

### 当测试判别位的P2为1且IntR为0时，表示没有中断请求且当前处于最后一条微指令，要跳转到取指微指令的地址，因此输出的多路选择器控制信号应为4，表示取指微指令地址。

1. 根据上述分析填写4号EXCEL表格中的组合逻辑真值表，如下图1-9所示。

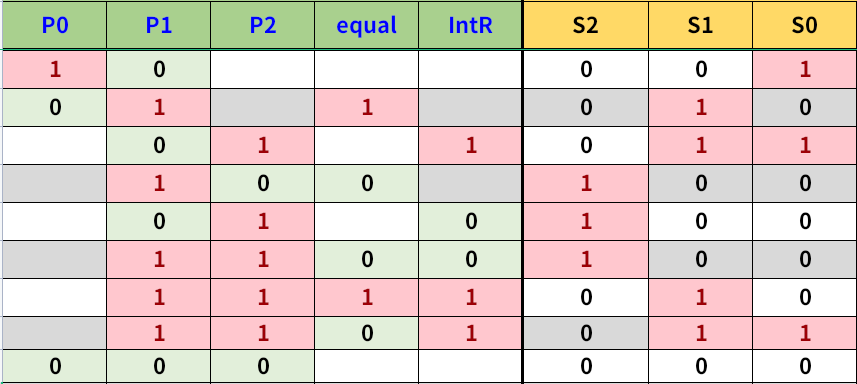


图1-9 条件判别测试逻辑真值表

1. 利用EXCEL自动生成的逻辑代数关系式在Logisim中自动生成电路图。生成的电路图如下图1-10所示。

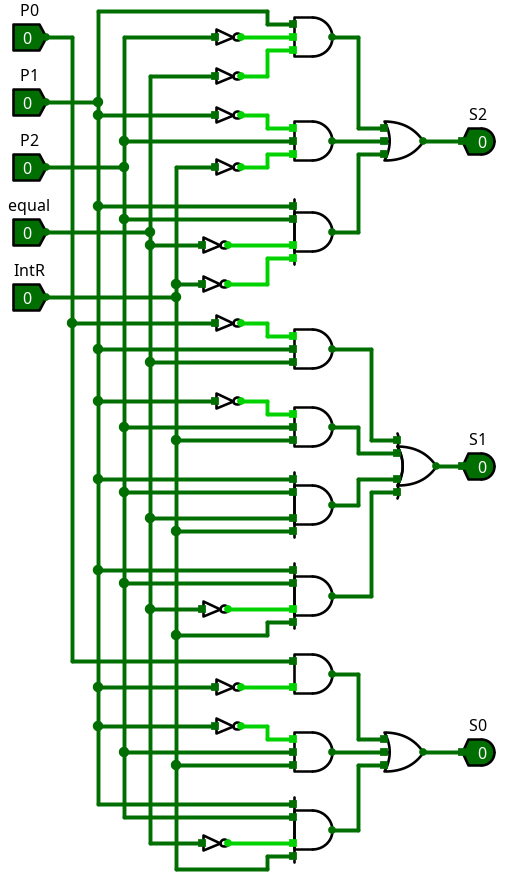


图1-10 条件判别测试逻辑电路

### 实现支持中断的微程序控制器

1. 将各个入口地址与多路选择器进行连接。其中顺序地址为mAddr+1的结果；入口地址为指令译码信号根据指令进行微地址转移的结果；BEQ分支的入口地址为常量16，即是十六进制下的0x10；中断处理程序入口地址为26，即十六进制下的0x1a，取值微指令的入口地址为0，即是十六进制下的0x00。电路连接如下图1-11所示。

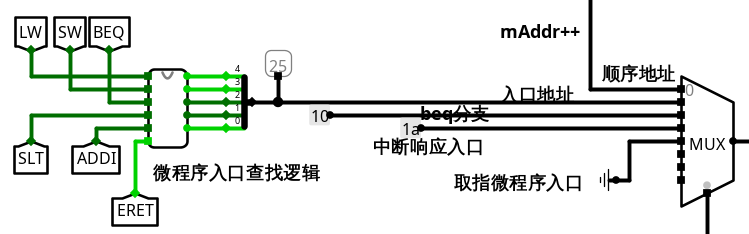


图1-11 微地址选择电路

1. 将判别测试逻辑输出的多路选择器控制信号连接到多路选择器上。电路连接如下图1-12所示。

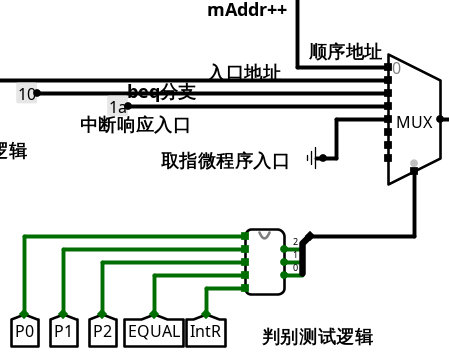


图1-12 微地址选择控制信号

1. 根据各个微指令需要的控制信号填写6号EXCEL表格中每个微指令地址对应的微指令控制信号，并自动生成相应的微指令。将微指令复制到控制寄存器中即可。微地址对应的微指令在EXECL表格中如下图1-13所示。

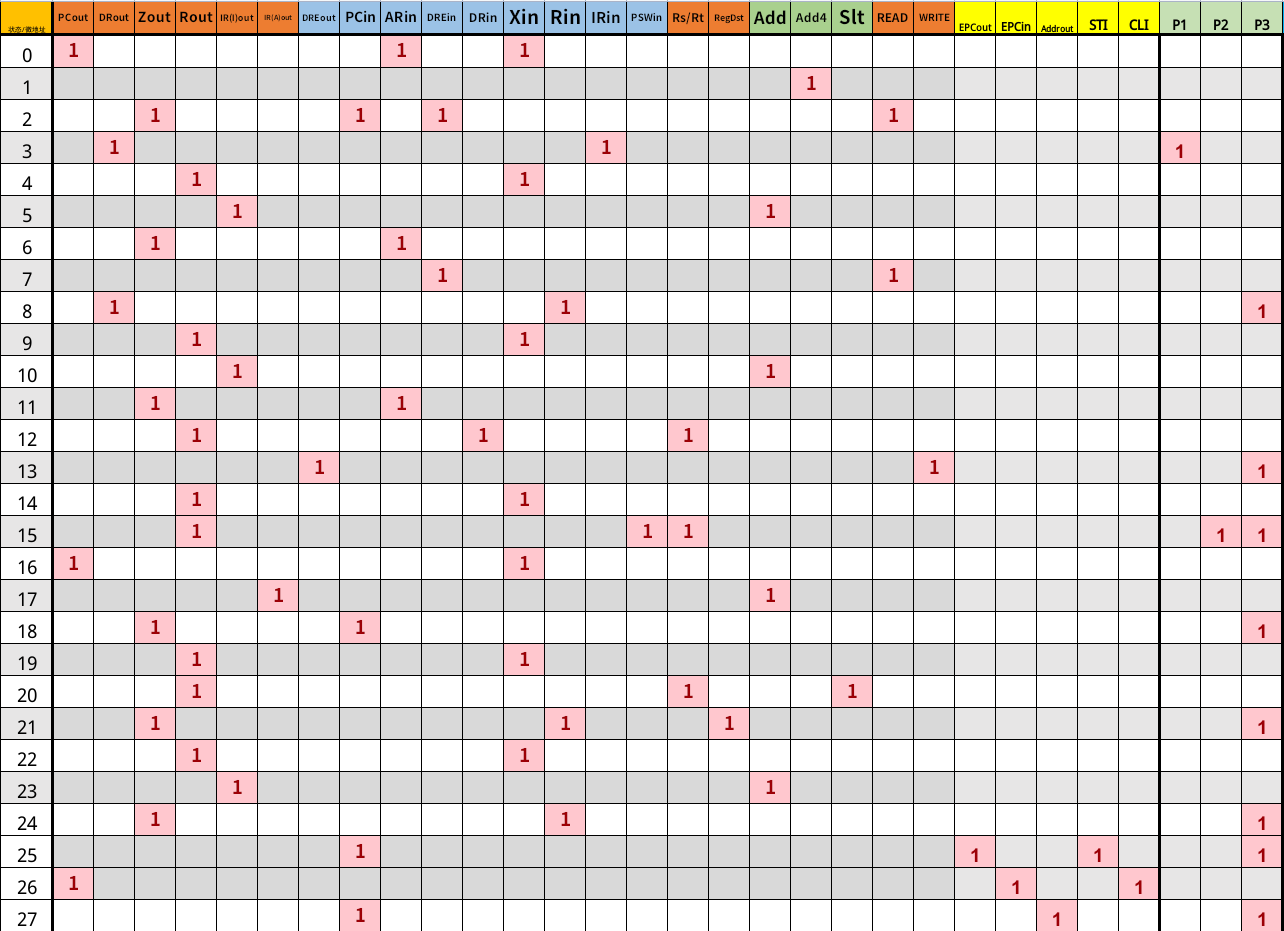


图1-13 微地址对应的微指令控制信号

### 实现支持中断的微程序单总线CPU

1. 使用中断控制器输出的IntR信号以及中断使能寄存器的读取端口通过一个与门，与门的输出端口即中断请求信号。注意，由于中断使能寄存器是关中断时为1，因此在输入到与门之前需要先通过一个非门进行去反。最终的逻辑为：当开中断且IntR为1时，中断请求到来。电路图如下图1-14所示。

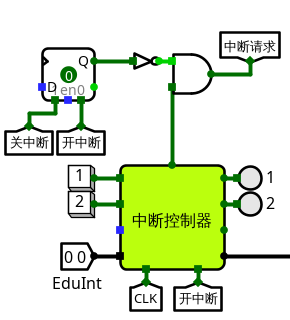


图1-14中断请求逻辑

1. 使用MARS汇编器查看sort-5-int.asm中两个中断处理程序对应的入口地址。这两个中断处理程序对应的入口地址如下图1-15所示。



图1-15中断处理程序入口地址

1. 使用多路选择器将中断控制器提供的中断号IntNo转化成中断处理程序的入口地址，并将中断处理程序的入口地址输入到内部总线里。注意，在中断地址输入到内部总线前要经过一个三态门，并使用Addrout信号来控制，这是因为当Addrout信号为 0时，程序不需要执行中断处理，因此中断处理程序的入口地址不能连接到内部总线中。选择中断处理程序入口地址的电路如下图1-16所示。

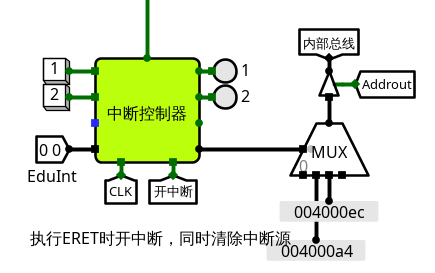


图1-16中断处理程序入口地址选择电路

1. 将内部总线连接到断点寄存器EPC的输入和输出端口中。在EPC的输入端与输出端各连接一个三态门，并在输入端使用EPCin信号、在输出端使用EPCout信号控制三态门。这是因为当CPU没有执行中断处理时不需要将内部总线中保存的汇编指令地址输入到EPC中，也不需要从EPC中读取汇编指令断点地址。断点寄存器的连接电路如下图1-17所示。

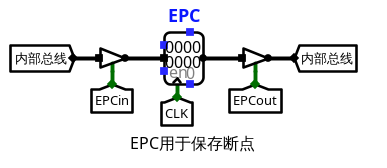


图1-17断点寄存器的连接

### 实现支持中断的硬布线控制器状态机

1. 硬布线控制器状态机通过输入信号的刺激将现态转化为相应的次态。
2. 当现态为S3时：若
3. 输入信号为LW，则次态为S4；
4. 输入信号为SW，则次态为S9；
5. 输入信号为BEQ，则次态为S14；
6. 输入信号为SLT，则次态为S19；
7. 输入信号为ADDI，则次态为S22；
8. 输入信号为ERET，则次态为S25；
9. 当现态为S8、S18、S21、S24、S25时（微程序最后一条指令）：若
   1. 输入信号中IR为0，则次态为0；
   2. 输入信号中IR为1，则次态为26；
10. 当现态为S15时：若
    1. 输入信号中equal为1，则次态为16；
    2. 输入信号中equal为0，IR为1，则次态为26；
    3. 输入信号中equal为0，IR为0，则次态为0；
11. 当现态为其他情况时，次态=现态+1。
12. 根据上述分析填写5号EXCEL表格，利用EXCEL自动生成的逻辑表达式在Logisim中生成相应的电路图。电路图如下图1-18所示。

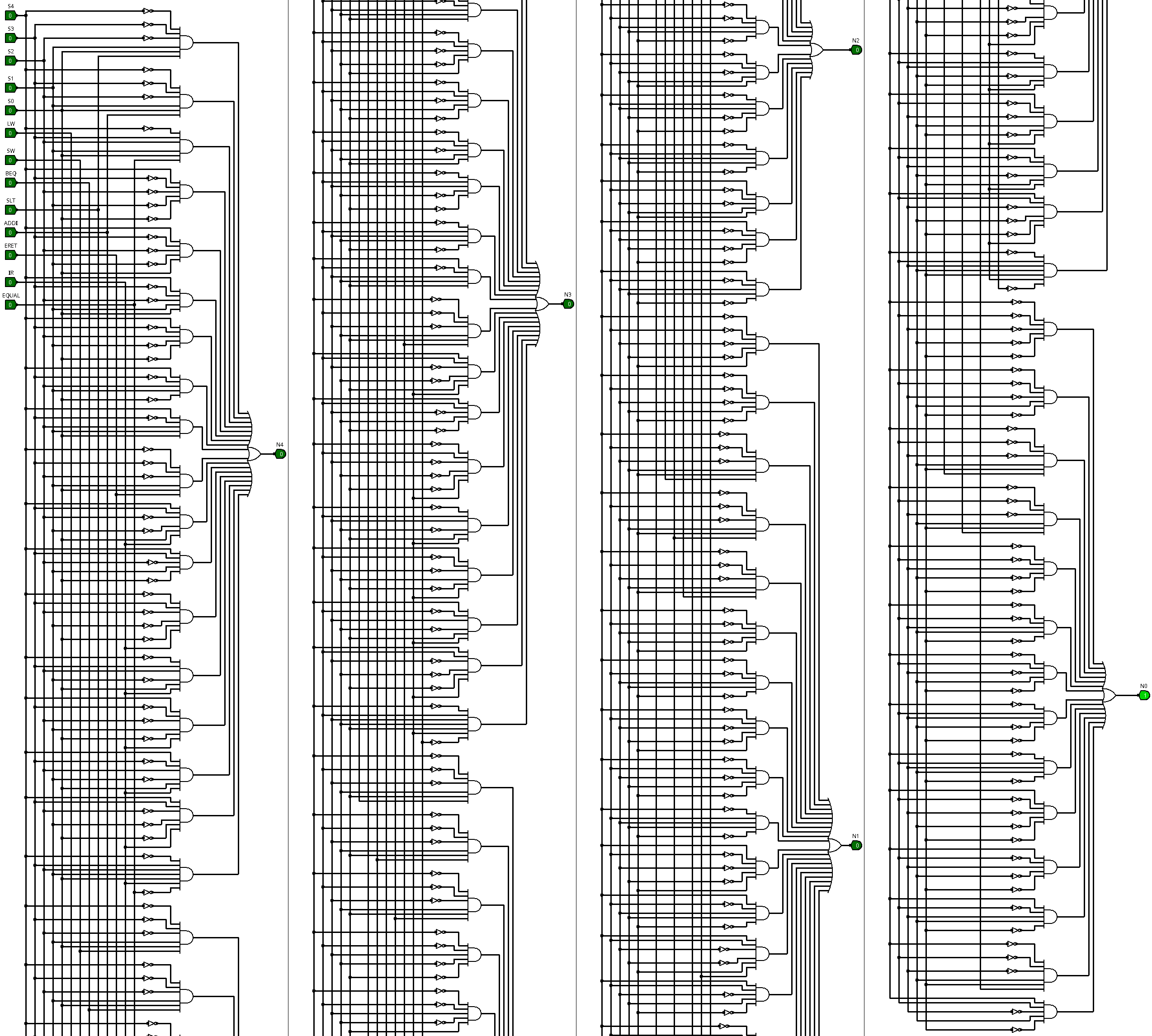


图1-18硬布线控制器状态机电路图

### 实现支持中断的硬布线控制器

1. 将当前状态status与控制信输入到硬布线控制器状态机中，再将生成的次态输入到次态寄存器的输入端中即可。实现的电路图如下图1-19所示。

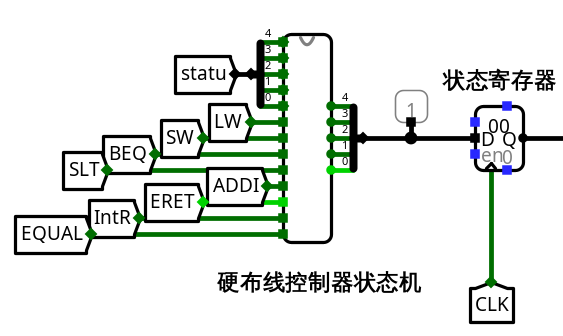


图1-19使用状态机生成次态

1. 将1.3.4中填写好的6号EXCEL表格中的微指令复制到控制存储器中。

## 故障与调试

### 微地址寄存器触发方式设置错误

**故障现象：**在头歌测试平台中错误总是奇数个指令发生错误。并且发生错误时总线的值总是为下一次时钟周期的总线的值。

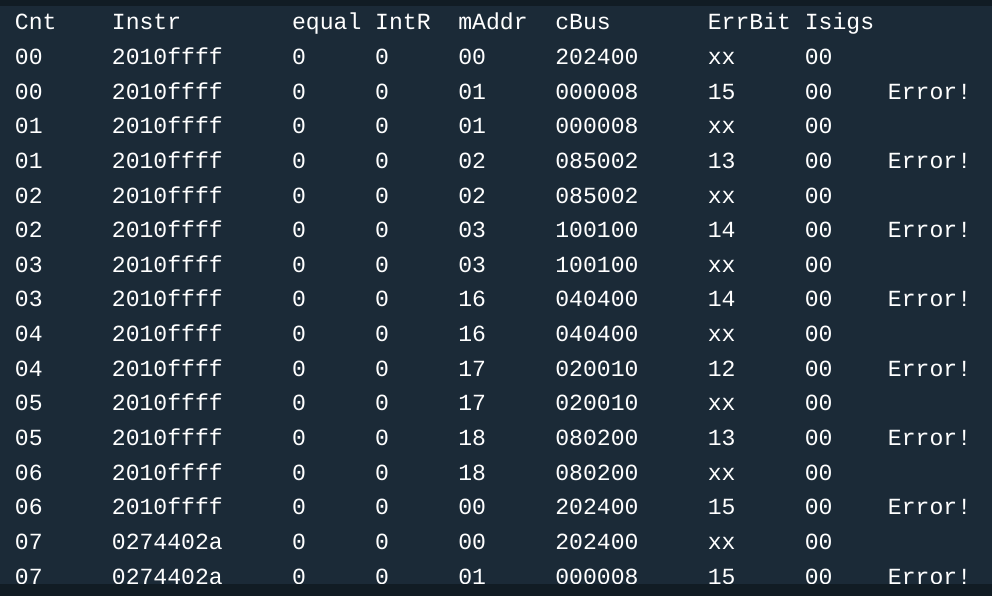


图1-20触发方式错误故障现象

**原因分析：**如图1-20，在发生错误的测试样例中，总线的值总是与下一次总线的值相同，考虑是总线的值改变时机不正确导致的。进一步分析发现问题在于微地址寄存器触发过早的问题。

**解决方案：**将微地址寄存器的触发方式从时钟上升沿触发改为下降沿触发，即可解决总线的值过早改变的问题。如图1-21所示。

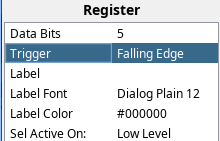


图1-21设置下降沿触发

### 判别测试逻辑输出引脚连接错误

**故障现象：**在头歌测试平台中进行测试，发现当mAddr只有0, 1, 2, 3四种取值。

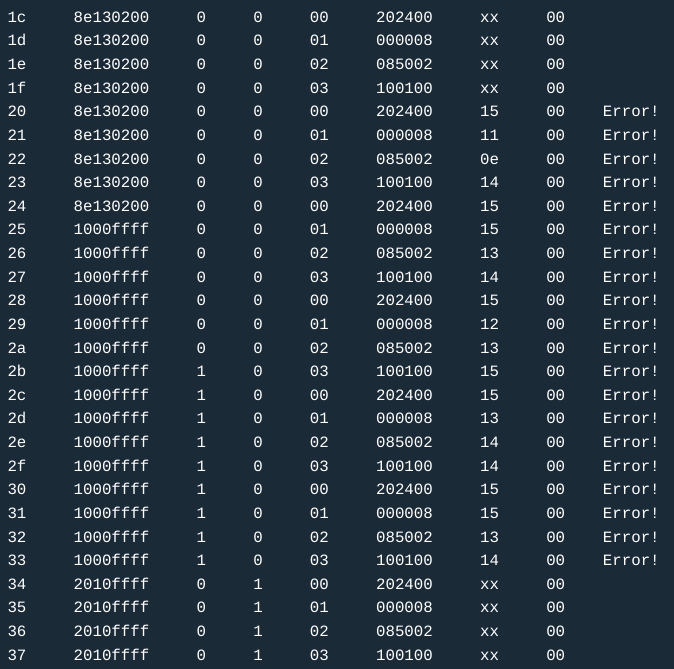


图1-22判别测试逻辑输出引脚连接错误

**原因分析：**如图1-22，程序的微地址mAddr总是在0, 1, 2, 3之间循环变化，无法到达其他的状态，因此考虑程序的状态图在0, 1, 2, 3之间有一个闭环。mAddr地址的选择是由判别测试逻辑进行控制的，先考虑判别测试逻辑发生了逻辑错误。但是由于判别测试逻辑在头歌平台中可以正常测试通过，因此进一步考虑判别测试逻辑与多路选择器之间的线路故障。经排查，问题在于判别测试逻辑输出端口连接的错误。由于判别测试逻辑一共有三个输出端作为多路选择器的控制信号，因此容易发生端口顺序连接混乱的问题，如下图1-23所示，S2端口连接到了分线器的0号引脚。

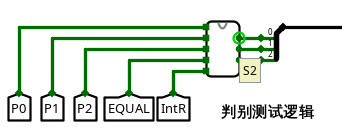


图1-23判别测试逻辑输出引脚连接错误

**解决方案：**将分线器的引脚从上到下分别设置为2, 1, 0。如图1-24所示。

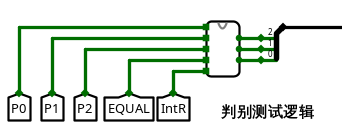


图1-24判别测试逻辑输出引脚连接错误修正

## 测试与分析

### 基于微程序控制器实现的支持中断的CPU执行sort-5-int.hex

1. 将支持中断的单总线CPU中的控制器换成微程序控制器。如下图所示。

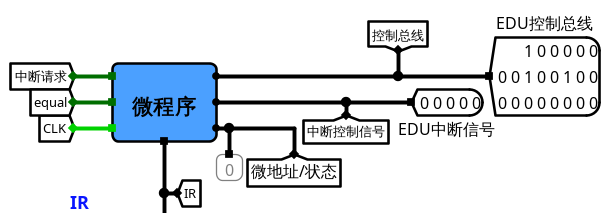


图1-25微程序控制器

1. 执行sort-5-int.hex程序后，内存中的内容如下图1-26所示。在0x80处开始的7个字节出现有符号降序排序。

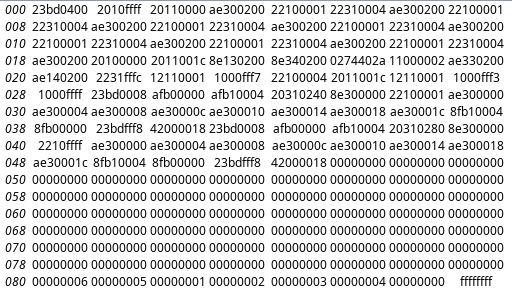


图1-26执行后内存中的内容

### 基于硬布线控制器实现的支持中断的CPU执行sort-5-int.hex

1. 将支持中断的单总线CPU中的控制器换成硬布线控制器。如下图所示。

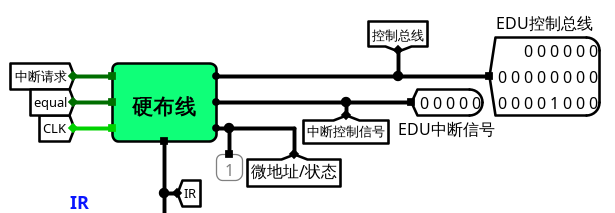


图1-27硬布线控制器

1. 执行sort-5-int.hex程序后，内存中的内容如下图1-28所示。在0x80处开始的7个字节出现有符号降序排序。

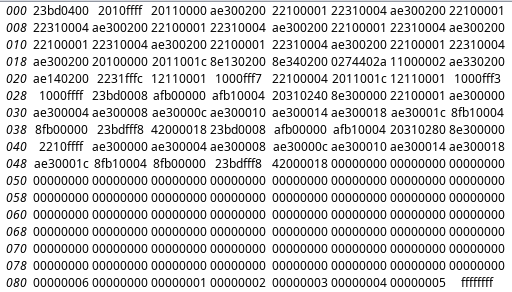


图1-28执行后内存中的内容

## 实验总结

条目式给出3次实验的主要工作，采用动宾结构，本次实验主要完成了如下几点工作：

1. 完成方案总结（自行修订扩充）（自行修订扩充）（自行修订扩充）（自行修订扩充）（自行修订扩充）。
2. 功能总结（自行修订扩充）（自行修订扩充）（自行修订扩充）（自行修订扩充）（自行修订扩充）。
3. 其他需要总结的内容，（自行修订扩充）。

## 实验心得

1. 在本次实验中，我对Logisim的使用相比之前的数字逻辑电路实验更加熟练了，学会了Logisim的一些高级用法，掌握了使用EXCEL填写真值表并自动转化成电路的方法。
2. 在本次实验中，我对计算机数据的表示、运算器的组成原理、存储系统的组成、单总线CPU的构成以及中断机制的实现都有了更深的理解。
3. 在本次实验中，我对在理论课中所学习的知识有了更深的理解，同时也将理论课中所学用到实践中来，这让我的知识不仅仅停留在纸面，而是可以在实践中运用了。
4. 在实验的过程中我也遇到了不少问题，有部分问题还十分难以发现。在解决问题的过程中，我动手实践的能力得到进一步地增强。
5. 总的来说，本次实验中我结合对理论知识的理解进行动手实践，这使我对计算机底层构造以及实现有了一个全面且深入的理解，这也将会为我以后深入学习计算机专业的相关知识打下坚实的基础。

|  |
| --- |
| 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: 郭德纲 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） | 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:** |