***2019***



**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： | CS1703 |
| 学 号： | U201714660 |
| 姓 名： | 饶鸿斌 |
| 电 话： | 15070357385 |
| 邮 件： | [767197265@qq.com](mailto:767197265@qq.com) |
| 完成日期： | 2019-11-27 |



目 录

[1 CPU设计实验 3](#_Toc25693728)

[1.1 设计要求 3](#_Toc25693729)

[1.2 方案设计 4](#_Toc25693730)

[1.3 实验步骤 5](#_Toc25693731)

[1.4 故障与调试 5](#_Toc25693732)

[1.5 测试与分析 6](#_Toc25693733)

[2 总结与心得 8](#_Toc25693734)

[2.1 实验总结 8](#_Toc25693735)

[2.2 实验心得 8](#_Toc25693736)

[参考文献 9](#_Toc25693737)

# 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。设计的CPU支持处理给定的指令集中列出的所有指令。指令见图1.1。



图1.1 指令集

## 方案设计

### 单周期MIPS（硬布线）

单周期MIPS硬布线的设计，我分了几个模块：

1、一个模块是指令计数器PC模块，针对PC的读取。其中包括了正常情况下PC+4的操作以及在跳转指令下PC+4的基础上加上偏移量。在设计的时候，我使用了一个数据选择器来选择PC+4还是跳转指令。选择的信号是单周期硬布线控制器的Beq和Bne端的输出。即当指令为Beq或者Bne的时候，给PC端的数据选择器一个选择信号。

2、一个模块是单周期硬布线控制器模块。该模块针对指令存储器中的op和func位产生相应的指令译码逻辑和ALU控制器逻辑，并且输出相应的控制信号。

3、一个模块是寄存器组和ALU控制器，该模块实现了对Rt、Rs、Rd的运算。此模块由老师提供。

4、一个模块是数据存储器，该数据存储器接收来自ALU运算器的I型指令lw、sw访问地址，输出相应地址的数据。该模块的作用是为了让所有指令都在一个时钟周期之内完成（同时完成对数据和指令的操作）。

5、其余的硬件设施：为了实现R型指令，我用了数据选择器选择Rd输入寄存器组。符号扩展器是为了便于I型指令立即数由16位扩展为32位和条件跳转指令中16位偏移地址的扩展。由于操作数和偏移量可正可负，需要采取符号扩展。

### 多周期MIPS微程序和硬布线

多周期MIPS微程序和硬布线的模块如下：

1、一个模块是指令计算器PC模块，该同单周期MIPS硬布线PC模块。

2、一个模块是多周期MIPS控制器模块。为了实现多周期下CPU不同指令对应不同的时钟周期数，每一个指令的执行拆解为三部分：取指、译码、执行。其中取指和译码两个阶段所有指令对应的数据通路是相同的。需要重点考虑的是指令的执行阶段。在指令的执行阶段需要做到：（1）按序进行；（2）根据指令本身功能对系统状态做出改变；

在多周期MIPS微程序控制器中，是根据指令进程微程序地址转移来输出不同微程序下的控制信号。在多周期MIPS硬布线控制器中，是根据状态机（现态和次态的转换）来输出不同状态下的控制信号。

3、一个模块是寄存器组和ALU控制器，该模块实现了对Rt、Rs、Rd的运算。此模块由老师提供。

4、一个模块是自定义寄存器模块。该模块包括了A、B、C三个寄存器，用于暂存RegiFile和ALU的数据输出。

5、其余的硬件设施：指令寄存器IR用于存放当前指令。数据存储器DR用于存放要操作的数据。符号扩展器，作用同单周期MIPS硬布线中的符号扩展器。

## 实验步骤

### 指令译码

将32位输入操作码用分线器接出。其中最高6位是操作码op，21-25位是rs寄存器编号，16-20位是tr寄存器编号，11-15是rd寄存器编号，0-5是功能码func。取其低16位作为I型指令中的立即数操作数。指令译码阶段，单周期和多周期是相同的。对应指令译码的线路连接如下图所示：

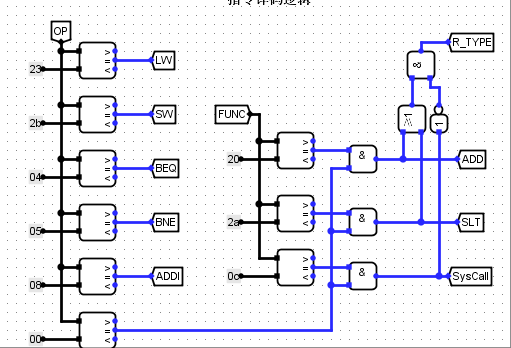


图1-3-1 指令译码

### 单周期硬布线控制器

控制器包括了指令译码部分。

1. ALU控制器逻辑

当OP=00时，选择func功能用来实现add和slt指令。ALU\_OP根据运算器规格可知当ALU\_OP为5时实现的是add，为11时实现的是slt。对应ALU控制器线路连接如下图所示：

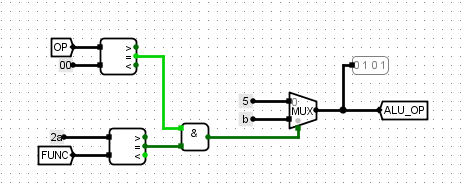


图1-3-2 ALU\_OP逻辑产生图

1. 控制信号输出逻辑

为了实现多指令的控制信号输出，我们需要利用译码电路生成的各指令译码信号，然后根据控制信号产生参见表将所有指令的控制信号逻辑或操作即可得到控制信号的逻辑表达式。

控制信号的产生参见表1-2-1

表1-2-1 控制信号说明以及产生条件

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **控制信号** | **信号说明** | **产生条件（信号为1）** |
| 1 | RegWrite | 寄存器写使能 | 寄存器写回信号 |
| 2 | MemWrite | 写内存控制信号 | sw指令 未单独设置MemRead信号 |
| 3 | AluOP | 运算器操作控制符（4位） | R型指令根据Func选择 |
| 4 | MemToReg | 寄存器写入数据来自存储器 | lw指令 |
| 5 | RegDst | 写入寄存器编号rt/rd选择 | R型指令 |
| 6 | AluSrcB | 运算器B输入选择 | lw指令，sw指令，立即数运算类指令 |
| 7 | SignedExt | 立即数符号扩展 | ADDI、ADDIU、SLTI指令 |
| 8 | JR | 寄存器跳转指令译码信号 | JR指令 |
| 9 | JAL | JAL指令译码信号 | JAL指令 ，选择寄存器写回编号，写回值 |
| 10 | JMP | 无条件分支控制信号 | J、JAL、JR指令，选择无条件分支地址 |
| 11 | Beq | Beq指令译码信号 | Beq指令，用于有条件分支控制 |
| 12 | Bne | Bne指令译码信号 | Bne指令，用于有条件分支控制 |
| 13 | Syscall | Syscall指令译码信号 | 根据$V0寄存器的值，决定是停机还是输出 |

根据上述的分析，我们可以生成单周期硬布线控制器，其电路连接如下图所示：

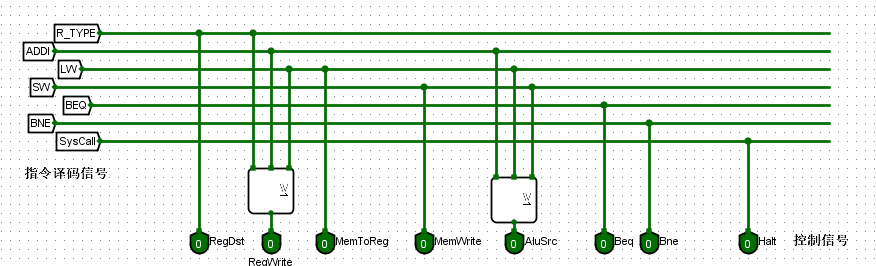


图1-3-3 单周期硬布线控制器电路图

### 单周期CPU总体结构图

CPU总体结构在上述说明的模块和器件的使用情况后，还需要进行数据通路的构建。为此，需要对主要功能部件的输入进行分析，主要功能部件输入表如下图所示：

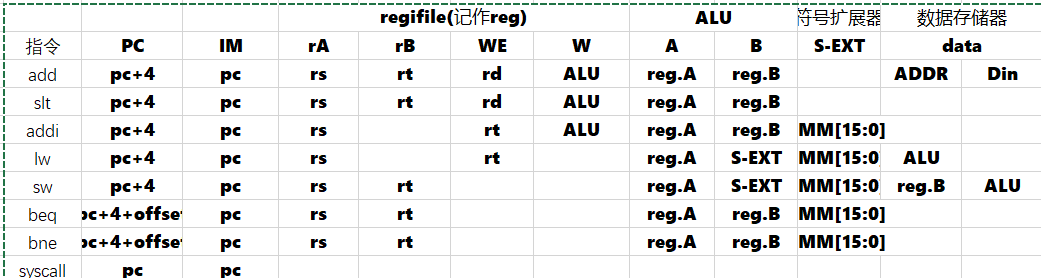


图1-3-4 单周期功能部件输入图

由上述分析，对相应的部件进行连线可得单周期硬布线控制器电路如下：

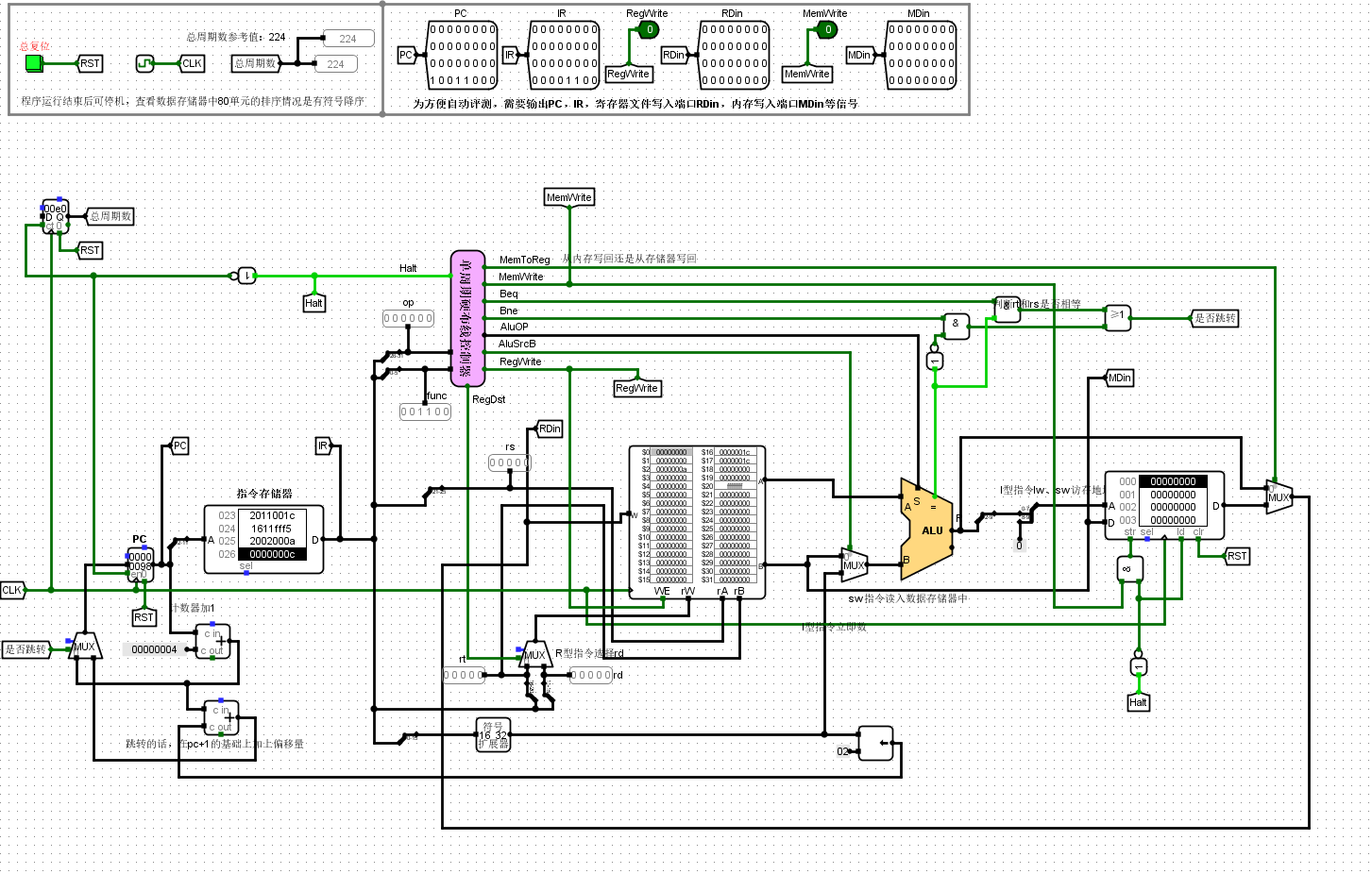


图1-3-5 单周期CPU总体结构图

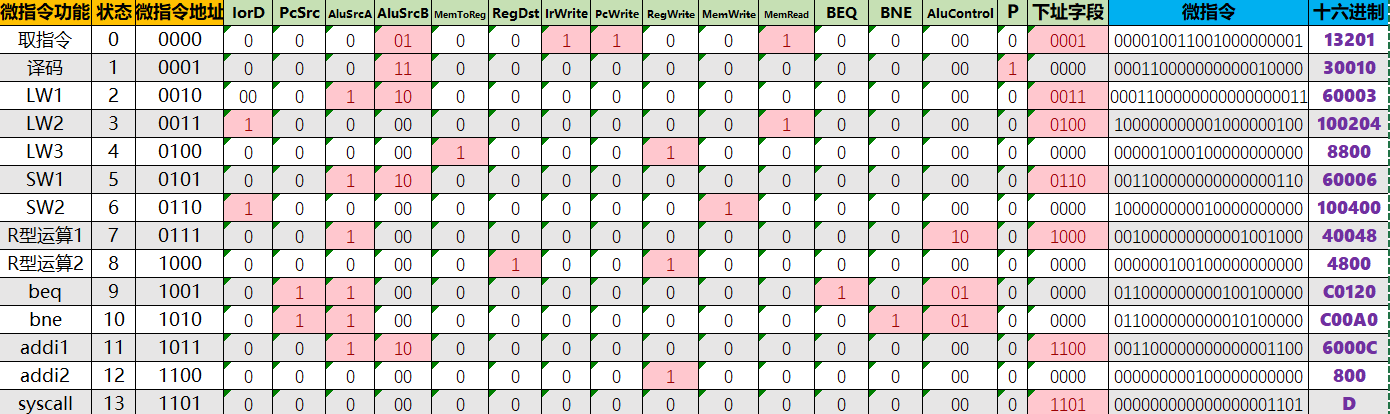
### 多周期控制信号

多周期CPU控制下不同指令对应不同的时钟周期数，因此不能一次性给出一条指令对应的所有控制信号。而每一条指令的执行可以分为三个阶段：取指、译码、执行。取值、译码部分的电路同单周期。结合8条指令的操作流程，可以得到所有指令不同阶段的控制信号表。



图1-3-6 8条指令的操作流程图

表1-3-1 指令控制信号表



控制信号说明如下图：



图1-3-7 控制信号说明图

### 多周期微程序控制器

1. 状态编码。8条指令一共13个执行阶段。当采取4位二进制编码表示微指令地址。如上表1-3-1所示。
2. 微指令地址转移逻辑。采用的是下址字段法，加上判断状态P。取指阶段P为1，表明控制存储器的地址由指令的第一个阶段的微程序地址给出，执行阶段P为0，表明下一条微指令地址由下址字段给出。根据表1-3-1给出的微指令功能和状态的对应关系，可以得出微程序地址的转移逻辑，如下图所示：



图1-3-8 微程序地址转移逻辑图

将对应的地址逻辑表达式生成输入到logsim中即可得到微程序地址转移逻辑电路，如下图所示：

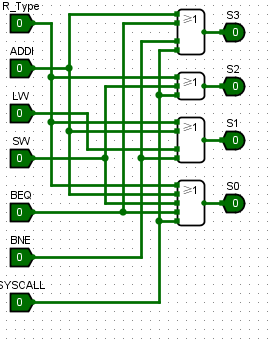


图1-3-9 微程序地址转移逻辑电路图

1. 控制存储器16进制微指令生成，由表1-3-1可知，8条指令执行需要的控制信号需要16位编码。并且由于一条指令对应多条微指令，需要完成下址字段。得到的微指令编码如表1-3-1所示。
2. 多周期微指令控制器综上所述，可以绘制出其电路图：

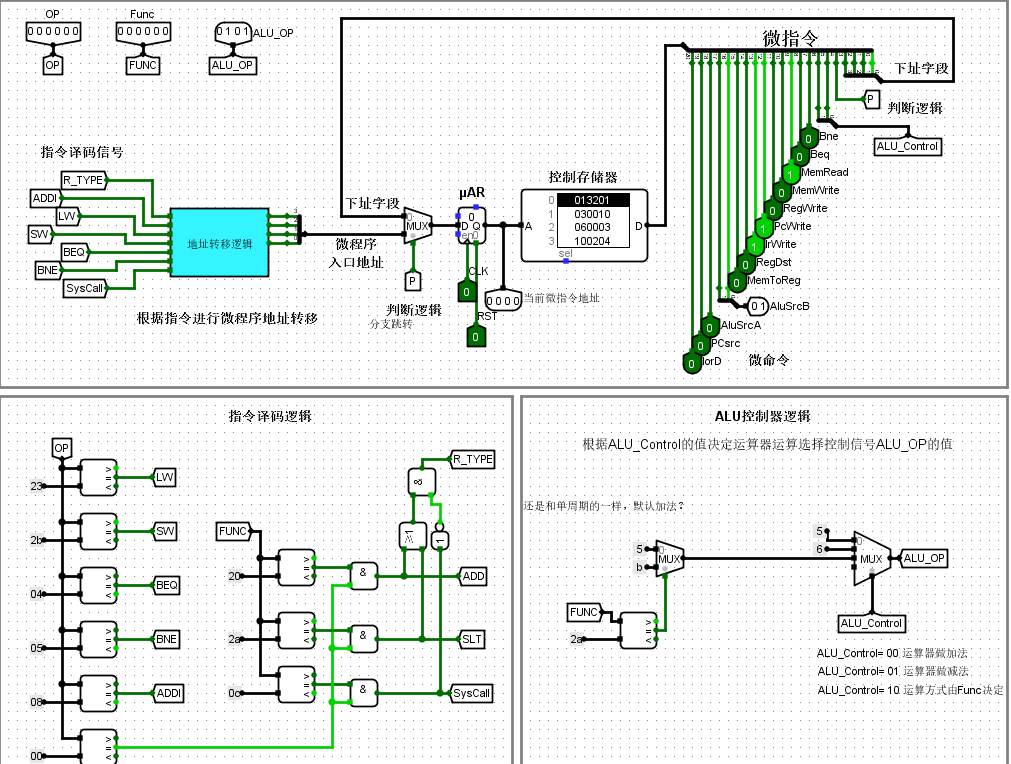


图1-3-10 微程序控制器电路图

### 多周期硬布线控制器

1. 状态编码，同微程序控制器编码。
2. 状态转移，多周期硬布线通过增加有限状态机FSM实现现态到次态的转换。其中，次态为下一条微指令在控制存储器中的地址。有限状态机FSM如下图所示：

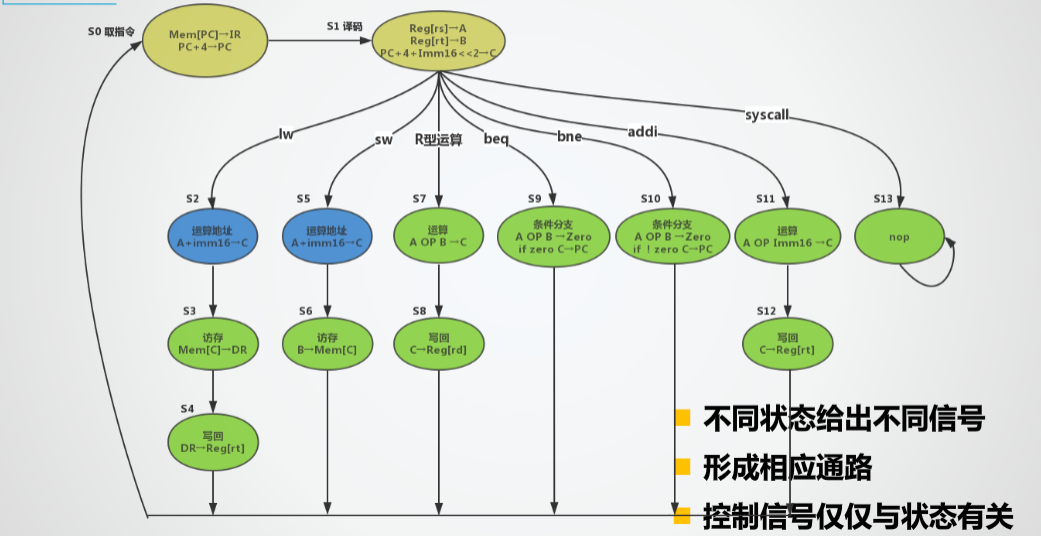
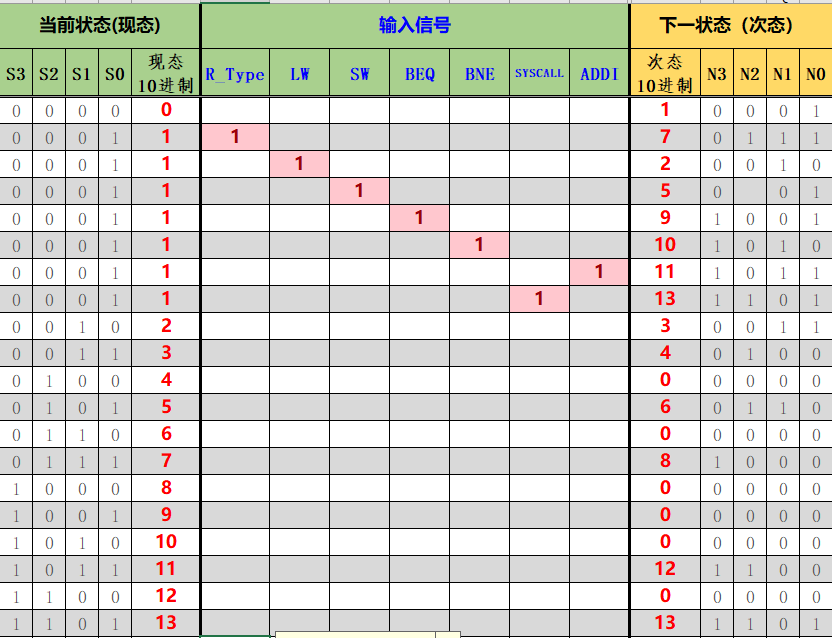


图1-3-11 有限状态机FSM示意图

1. 根据表1-3-1中微指令功能、状态、微指令地址的对应关系，可以得到FSM状态转换地址逻辑表：

表1-3-2 FSM状态转换地址逻辑表



1. 由于状态转移由FSM给出，故将微程序的控制器减去下址字段和P。和微程序控制器电路实现相似，可以得到多周期硬布线控制器的电路图：

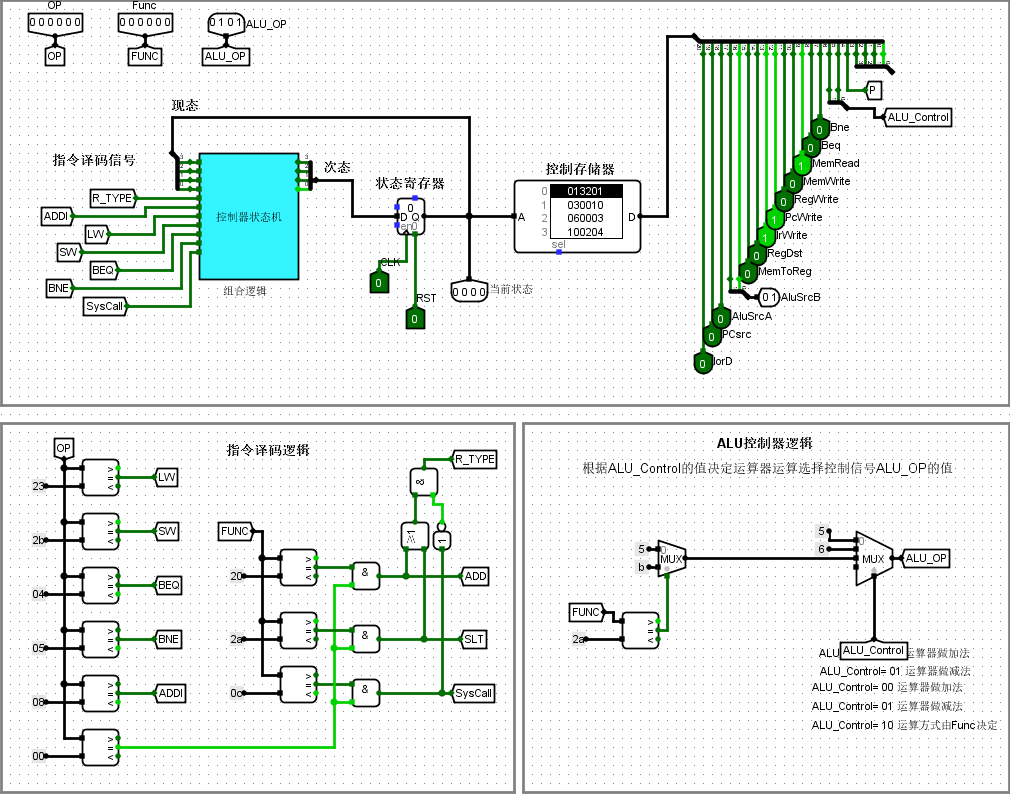


图1-3-11 多周期硬布线控制器电路图

### 多周期CPU总体结构图

由于多周期的部件在之前已经分析过了，存储器、Regfile、ALU等主要器件和单周期硬布线中的一致，故只需要构建数据通路连接即可。

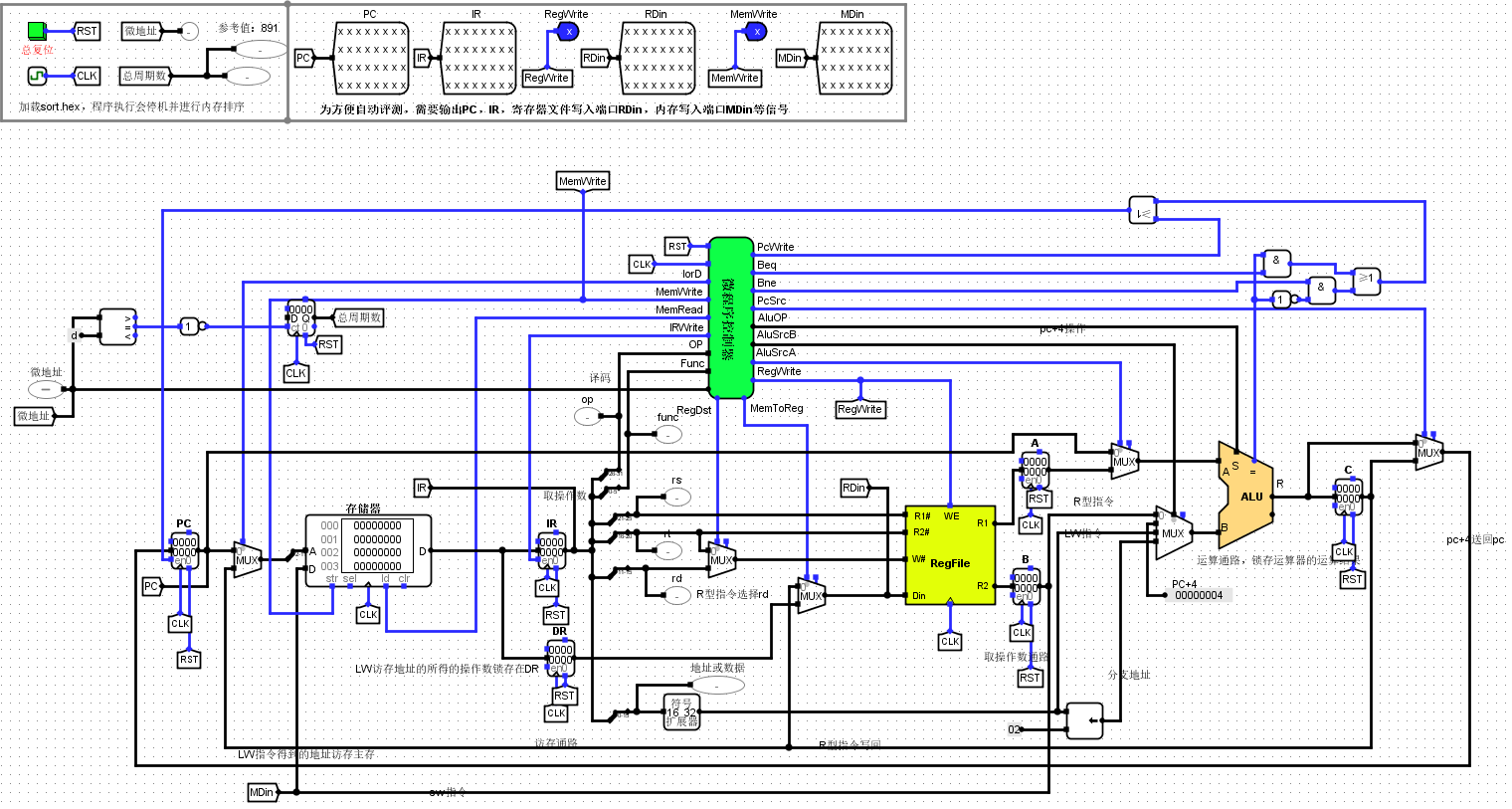


图1-3-12 多周期微程序CPU总体结构图

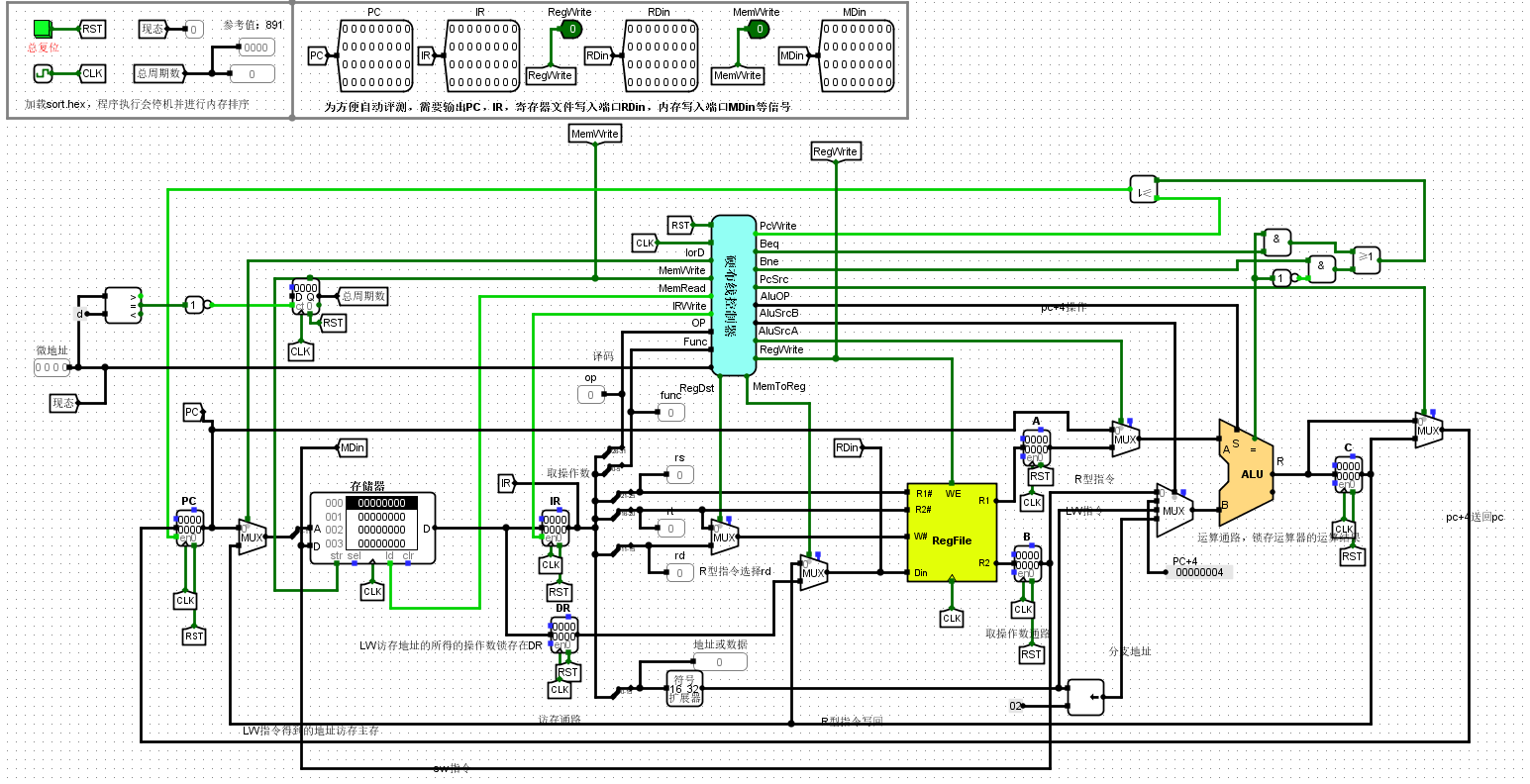


图1-3-13 多周期硬布线CPU总体结构图

## 故障与调试

### ALU控制器逻辑问题

**故障现象：**执行到第36个周期的时候，addi指令无法正确执行，下个周期的目标地址没有送入到PC中，导致系统卡住。

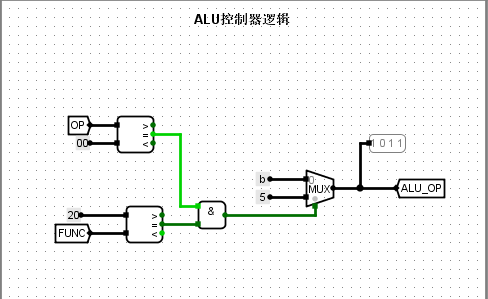


图 1-4-1 ALU控制器逻辑故障图

**原因分析：**如图1-4-1，当OP为00时根据func功能号选择相应的ALU\_OP。但是根据指令的执行顺序，在默认的情况下，ALU\_OP应该为add信号 让ALU实现两数相加的操作，而不应该默认为符号比较操作。

**解决方案：**修改ALU控制器逻辑，默认为加法操作。

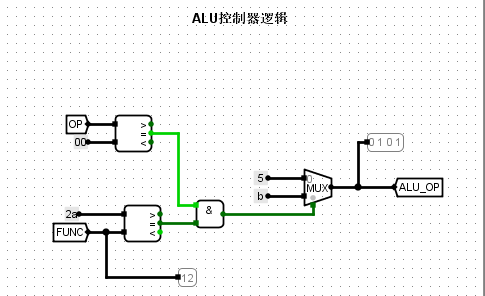


图 1-4-2 ALU控制器逻辑故障修改图

### 多周期CPU循环震荡问题

**故障现象：** 多周期CPU运行sort.hex程序时，遇到停机指令无法正常停机。PCEn使能信号产生震荡，pc循环。

**原因分析：**指令译码阶段出现错误。错误的将syscall指令当成了R型指令，没有区分导致控制器没有产生正确的PCWrite信号。

**解决方案：**区分syscall指令和R型指令，使得syscall指令和R型指令互斥。

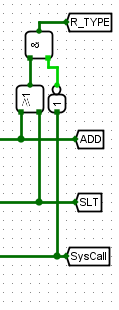


图 1-4-3 syscall指令无法执行故障修改图

## 测试与分析

### 单周期硬布线CPU执行sort.hex

1. 在数据存储器中，80号单元开始出现了6，5，4，3，2，1，-1的有符号降序数据。如下图所示：



图 1-5-1 单周期硬布线CPU数据存储器内存显示图

1. 时钟周期。执行完毕后，系统停机，sort.hex的时钟周期数为224。如下图所示：



图 1-5-2 单周期硬布线CPU系统停机时钟周期图

### 多周期CPU微程序执行sort.hex

1. 数据存储器在80号单元出现6，5，4，3，2，1，-1的有符号降序数据。如下图所示：

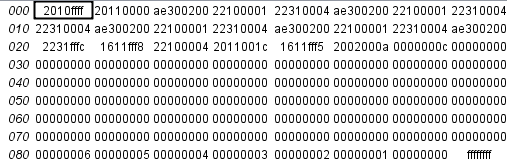


图 1-5-3 多周期微程序CPU存储内存显示图

1. 时钟周期。执行完毕后，系统停机，sort.hex的时钟周期数为891。如下图所示：

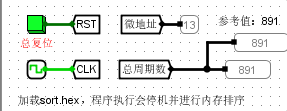


图 1-5-4 多周期微程序CPU系统停机时钟周期图

### 多周期CPU硬布线执行sort.hex

1. 数据存储器在80号单元出现6，5，4，3，2，1，-1的有符号降序数据。如下图所示：

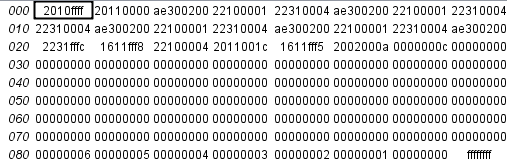


图 1-5-3 多周期硬布线CPU存储内存显示图

1. 时钟周期。执行完毕后，系统停机，sort.hex的时钟周期数为891。如下图所示：

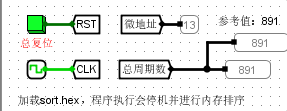


图 1-5-4 多周期硬布线CPU系统停机时钟周期图

# 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MIPS单周期硬布线CPU控制器。
2. 实现了MIPS多周期硬布线和微程序CPU控制器。
3. 实现了PC指令的顺序读取和跳转指令读取。
4. 实现了MIPS CPU数据通路综合，使得CPU可以完整支持MIPS的8条指令。
5. 实现了立即数扩展。
6. 实现了寄存器的读写。
7. 实现了数据存储器的读写。
8. 实现了周期数目的记录。
9. 实现了控制信号的生成。

## 实验心得

1. 实验加深了对logsim的使用，包括对故障电路的调试。
2. 加深了对CPU的理解。包括单周期和多周期，以及不同设计模式下CPU的运行机理和内部模块的协同工作。
3. 熟悉了MIPS核心指令，对其三大类型的指令的执行有了更深的理解。
4. 本次实验可以说是和课本同步的。在做实验的过程中，有很多不熟悉和一知半解的知识点。这给我的实验带去了很大的麻烦。但是在温习了老师的上课知识点结合慕课的学习，让我对CPU这一个章节的知识有了更深的学习。实验从侧面检测着学习的状况，可以让我可以及时的发现自己学习上不足和不够的地方。我觉得这点很好。
5. 本次实验唯一的小瑕疵在于实验文档不断更新，让我需要不断调整，打乱了原本的学习计划。当然，我也理解实验老师是希望让实验更加完善。希望在之后的实验中，可以及时发布相关的不同版本的差异，让学生们可以更加直观和及时的做出调整。
6. 最后，很感谢QQ群里不知疲倦回答同学们问题的老师和慕课对实验讲解的极为详细的授课老师。还有在我做错和有困惑时帮助我的同学！

# 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:** |