

**课 程 实 验 报 告**

**课程名称：** 计算机组成原理

专业班级： **CS1902**

学号： **I201920024**

姓名：木林

指导教师：谭志虎

报告日期： **2021**年**12**月**5**日

计算机科学与技术学院

**目录**

实验 CPU设计要求 1

1设计要求 1

1.1单总线结构现代时序CPU 1

1.2单总线结构现代时序CPU（带中断操作） 1

1.3单总线结构三级时序变长指令CPU 2

2.1单总线结构现代时序CPU 4

2.2单总线结构现代时序CPU（带中断操作） 9

2.3单总线结构三级时序变长指令CPU 14

# 实验 CPU设计要求

## 1设计要求

### 1.1单总线结构现代时序CPU

本实验要做的包括：

1. 指令译码器
2. 微程序入口查找逻辑
3. 微程序条件判别测试逻辑
4. 微程序地址转移逻辑
5. 微程序指令存储器
6. 时序产生器（硬币线有限状态机）
7. 硬布线控制信号产生器

图 1.1单总线结构现代时序总体结构图

### 1.2单总线结构现代时序CPU（带中断操作）

本实验要做的：

1. 指令译码器（带ERET指令）
2. 微程序入口查找逻辑
3. 微程序条件判别测试逻辑
4. 微程序地址转移逻辑
5. 微程序指令存储
6. 时序产生器（硬币线有限状态机）
7. 硬布线控制信号产生器

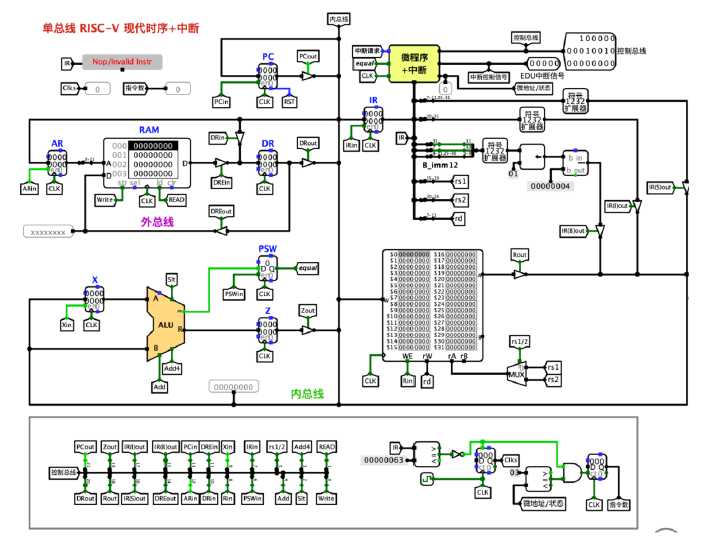


图 1.2单总线结构现代时序（带中断）总体结构图

### 1.3单总线结构三级时序变长指令CPU

本实验要做的：

1. 指令译码器
2. 硬币线时序发生器
3. 硬币线时序发生器输出逻辑
4. 硬币线控制器组合逻辑单位
5. 硬币线控制器

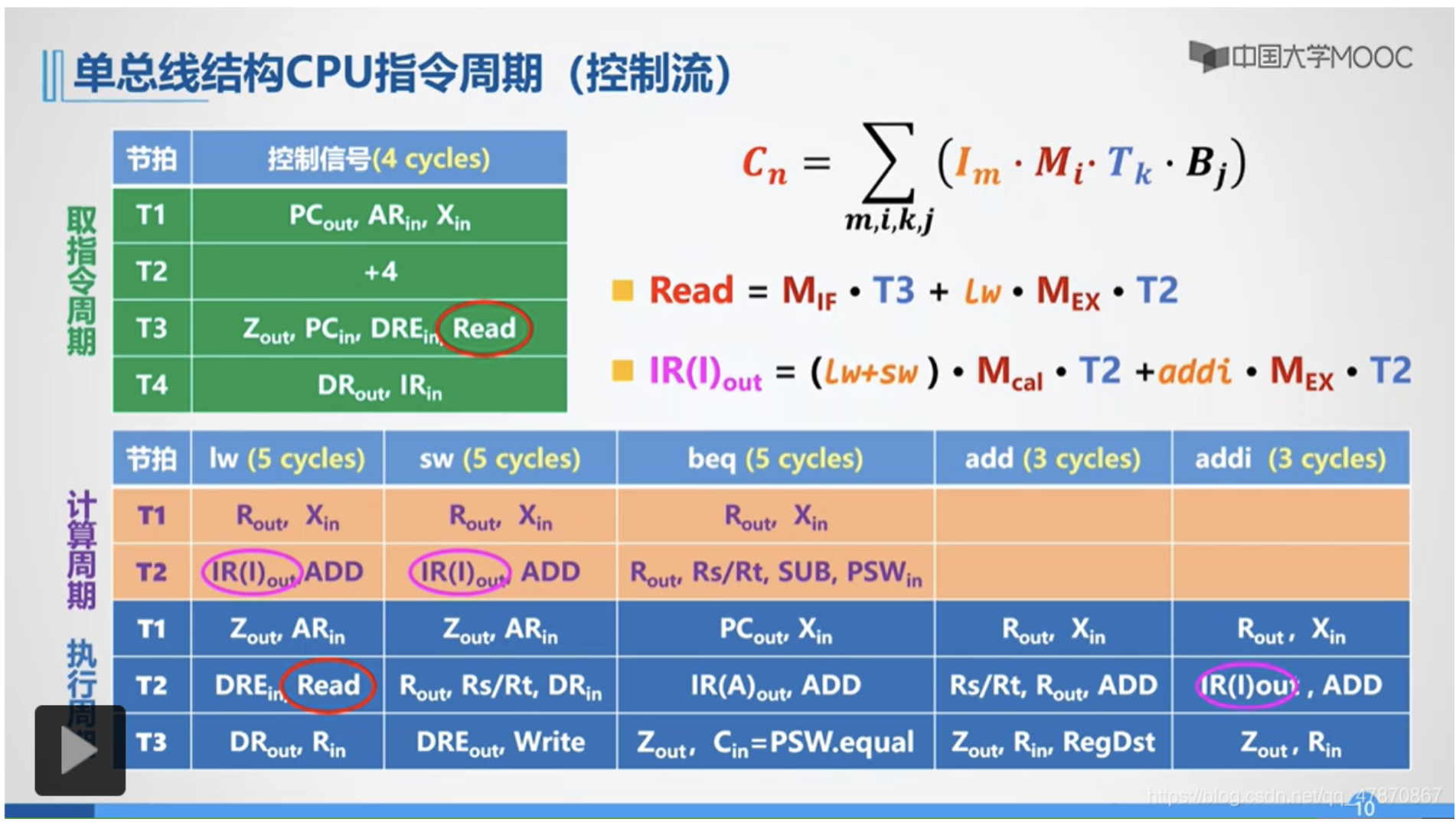


图1.3a中国大学mooc中的单总线结构三级时序变长指令CPU指令期

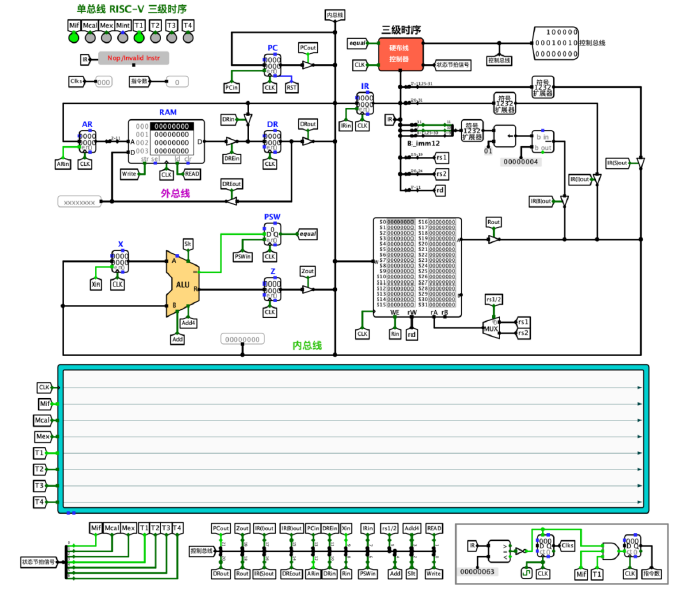


图 1.4单总线结构三级时序变长指令总体结构图

### 2.1单总线结构现代时序CPU

* 指令译码器

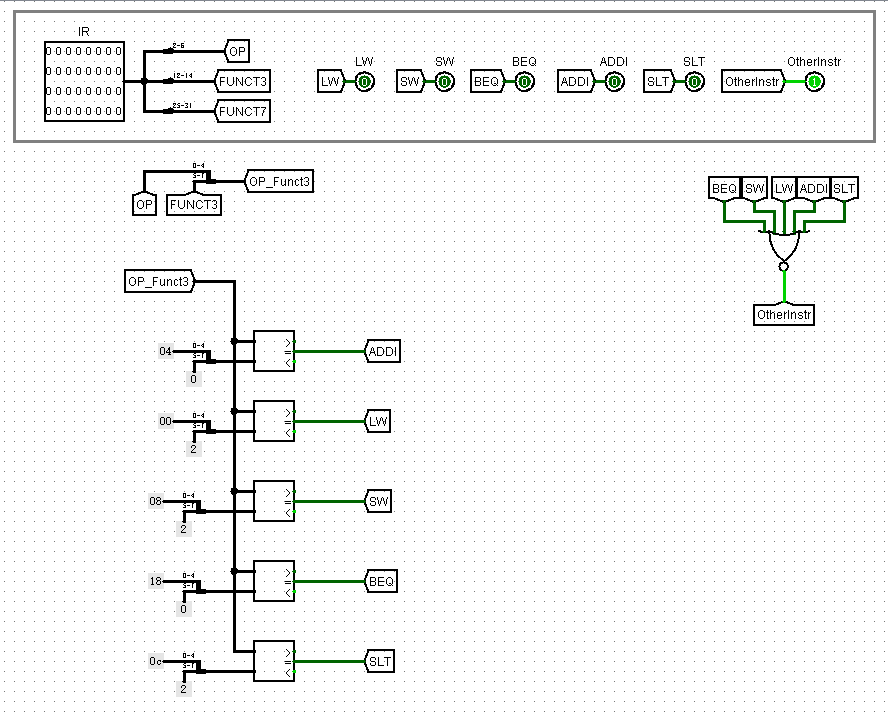


图 1.5 单总线结构现代时序指令译码器总体结构

图 1.5单总线结构现代时序指令译码器总体结构图

输入：32位指令IR

输出：指令译码器的结果：LW信号代表指令是否为LW指令，SW信号表表指令是否为SW指令，BEQ信号代表指令是否为BEQ指令，ADDI信号代表指令是否为ADDI指令，SLT信号代表指令是否为SLT指令

实现逻辑：将指令IR的2—6（opcode）和12-14（func3）位提取出来进行比对，用5个比较器来判断该指令的opcode和func3是否与标准指令匹配。

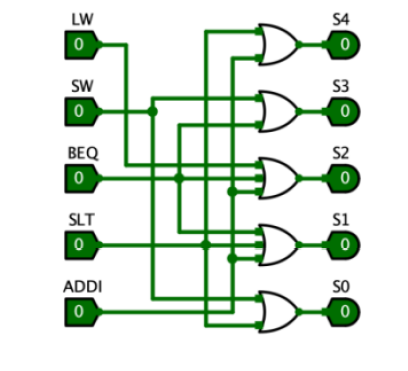
* ****

图 1.6单总线结构现代时序微程序入口查找逻辑总体结构图

输入：5个指令选择信号，代表译码出来的指令类型。

输出：微程序入口地址（S=[S4S3S2S1S0]）

实现逻辑：利用Excel表构造组合逻辑，然后在logism组合逻辑生成器生成电路。

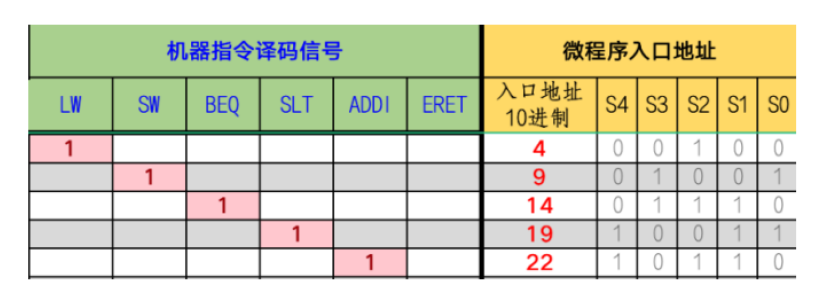


图 1.7单总线结构现代时序微程序入口查找逻辑组合逻辑设计表

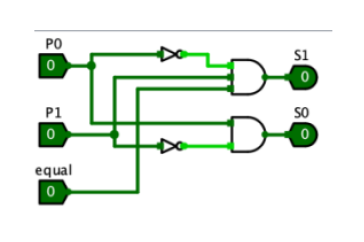
* 微程序条件判别逻辑

图 1.8单总线结构现代时序微程序条件判别逻辑组合总体结构图

实现逻辑：微程序流程控制信号P=[P1P0]和表示相等关系信号equal。

微程序地址转移控制信号(S=[S1S0]):S=0代表是下一条地址，S=1代表取入口地址，S=2代表是取BEQ指令且当前运算是相等的跳转地址。利用Excel表构造组合逻辑，然后在logism生成电路。

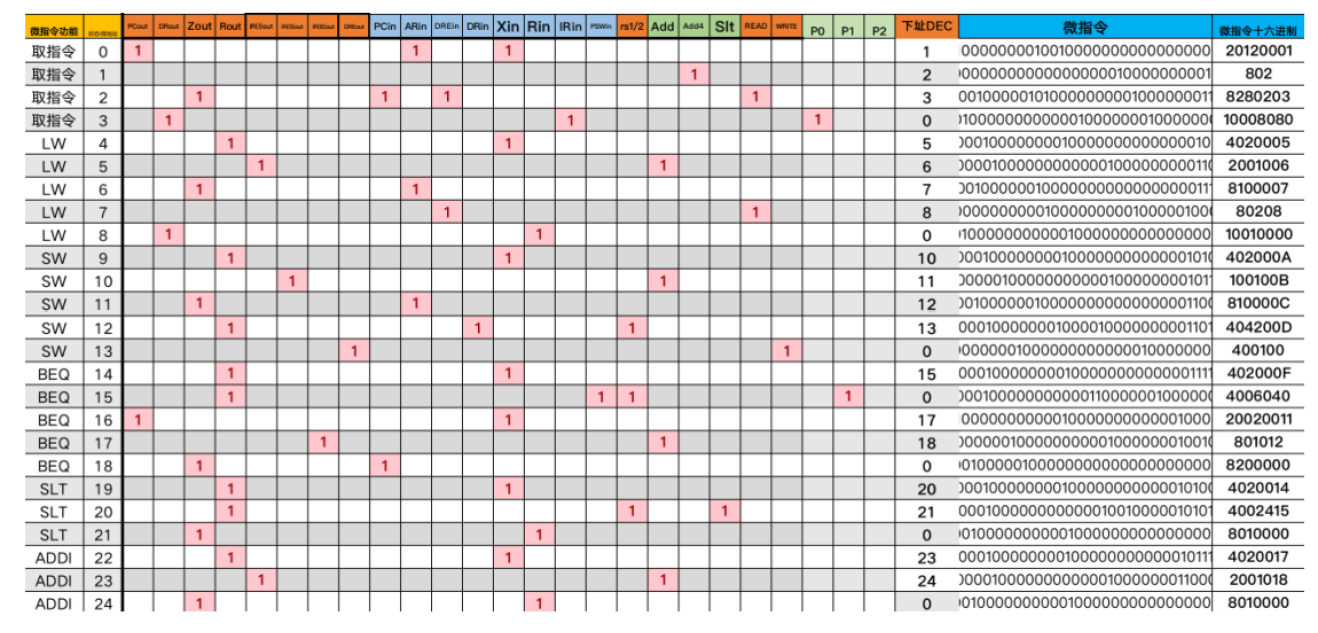
* ****控制寄存器

图 1.9单总线结构现代时序微程序控制寄存器内容

实现逻辑：微程序的地址对应硬布线的时序，我们只需要对照着硬布线的控制信号输出表填写控制信号，然后根据FSM的关系设置下一条微程序的地址，设计逻辑如下：

1. 在不需要跳转的微程序处，下一条微程序是本指令地址+1
2. 在取指令的最后一条微程序标记下一条微程序的地址是微程序入口生成地址逻辑结果。
3. 在每一条指令的最后一条微程序标记下，一条指令是取指令的第一条指令
4. 对于equal的判断：如果在S15条微程序的最后设置下一条指令与equal信号有关，交由微程序条件判别逻辑来判断，如果是相等的话(equal=1)，跳转到S16。如果不是，返回取指令的第一条指令。

* 微程序控制器

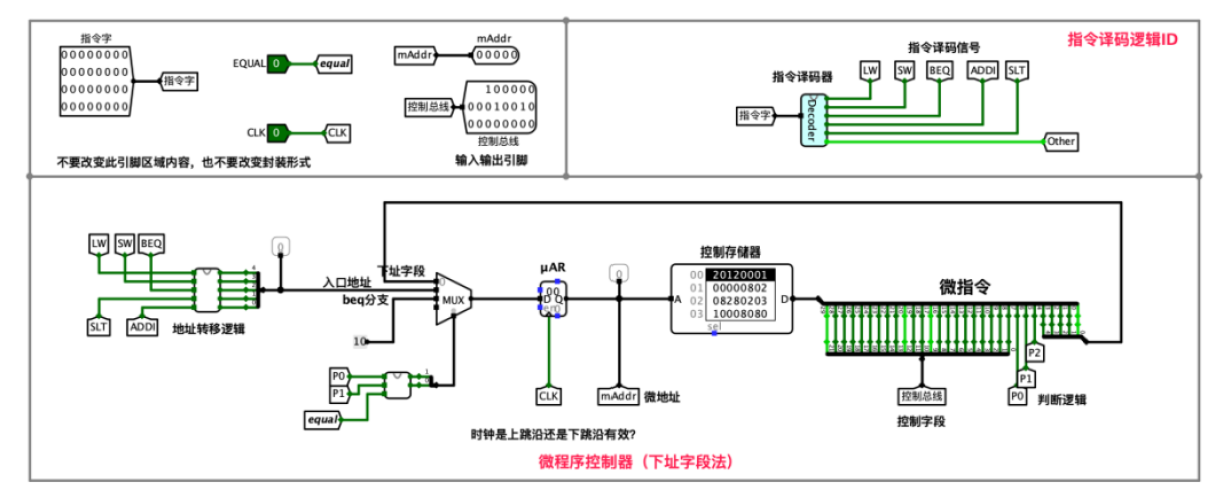


图 1.10单总线结构现代时序微程序控制器总体结构图

实现逻辑：其他功能模块已经在之前说明，这里最重要的就是下一条微程序的地址：需要根据微程序条件判别逻辑的结果来进行多路选择。如果是下一条地址就对应当前微程序的下地址字段。如果是入口地址就接入入口地址查找逻辑的输出。如果是beq分支就跳转到0x10。

* 硬布线时许产生器

实现逻辑：Excel生成组合逻辑，然后在logism生成电路

* 硬布线控制器

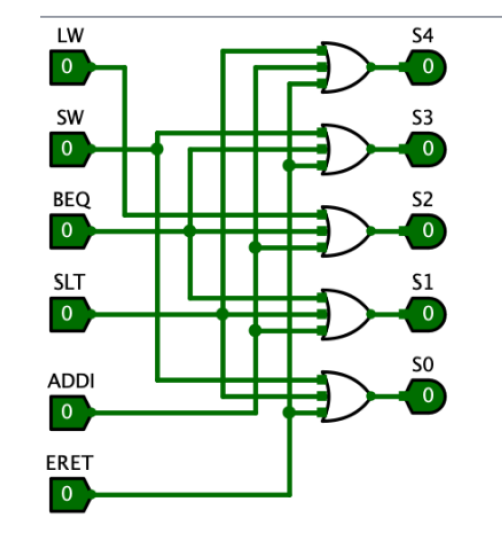
实现逻辑：主要的就是状态寄存器右边部分是当前状态，左边部分是次态，将当前状态和指令译码器的输出作为FSM的输入连线即可。因为硬布线的状态和微程序的地址是一一对应的，这个时候可以借用之前做的微程序控制存储器来读取控制总线输出。

### Picture 202.2单总线结构现代时序CPU（带中断操作）

图 1.11单总线结构现代时序硬布线控制器总体结构

* 指令译码器

与2.1的相同。略

* ****微程序入口查找逻辑

1图1.2单总线结构现代时序（带中断）微程序入口查找逻辑总体结构图

输入：6个指令选择信号，代表译码出来的指令类型。

输出：微程序入口地址（S=[S4S3S2S1S0]）

实现逻辑：用Excel构造组合逻辑，然后在logism生成电路

* 微程序条件判别逻辑

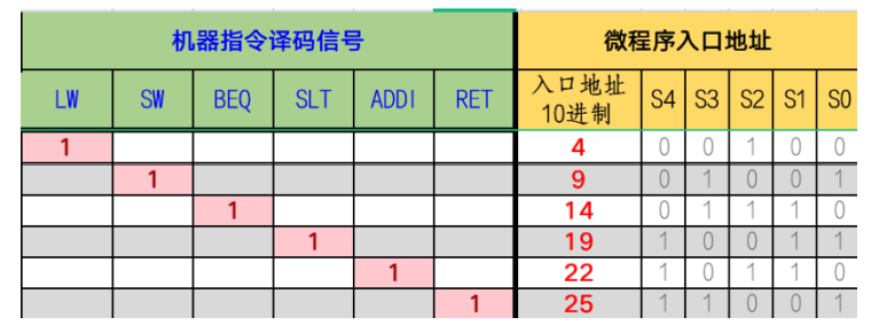


图1.13单总线结构现代时序（带中断）微程序入口查找逻辑组合设计表

实现逻辑：用Excel表构造组合逻辑，然后在logism生成电路。

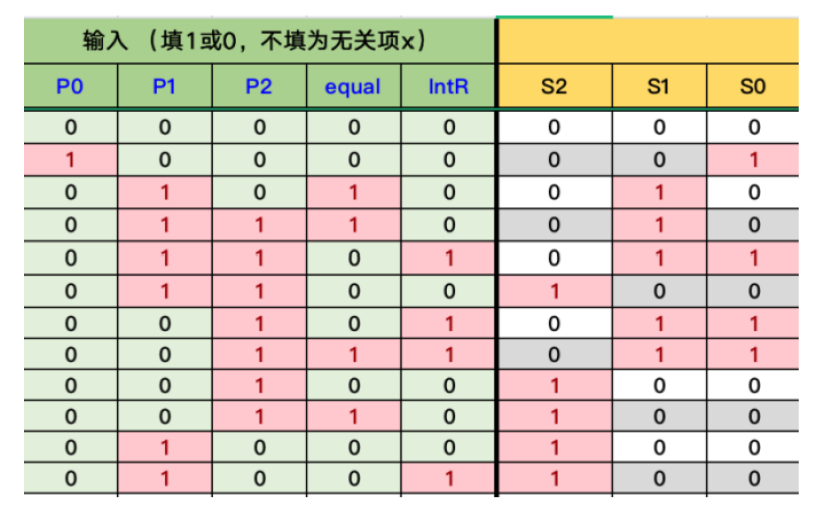


图1.14单总线结构现代时序（带中断）微程序条件判别逻辑组合逻辑设计表

* 控制寄存器

实现逻辑：微程序的地址对应硬布线的时序，我们只需要对照着硬布线的控制信号输出表填写控制信号，然后根据FSM关系设计流程控制信号P。

1. 当这个是取地址指令的最后一条微程序的时候，这个时候标记P0要找入口查找逻辑寻找入口地址。
2. 当这个是判断equal分支的时候，这个时候标记P1有可能要跳转到处理equal分支的微程序
3. 当这个是每一条指令的最后一条微程序的时候，标记P2代表结束。

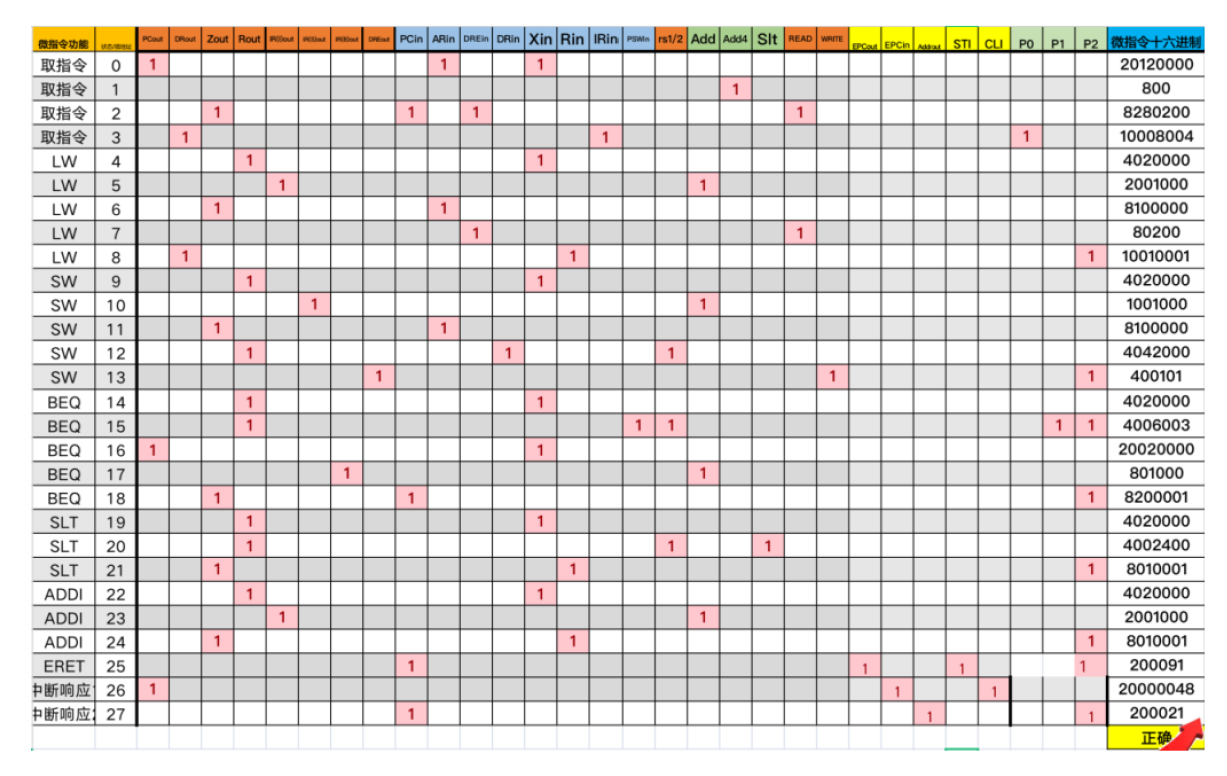


图1.15单总线结构现代时序（带中断）微程序控制寄存器内容

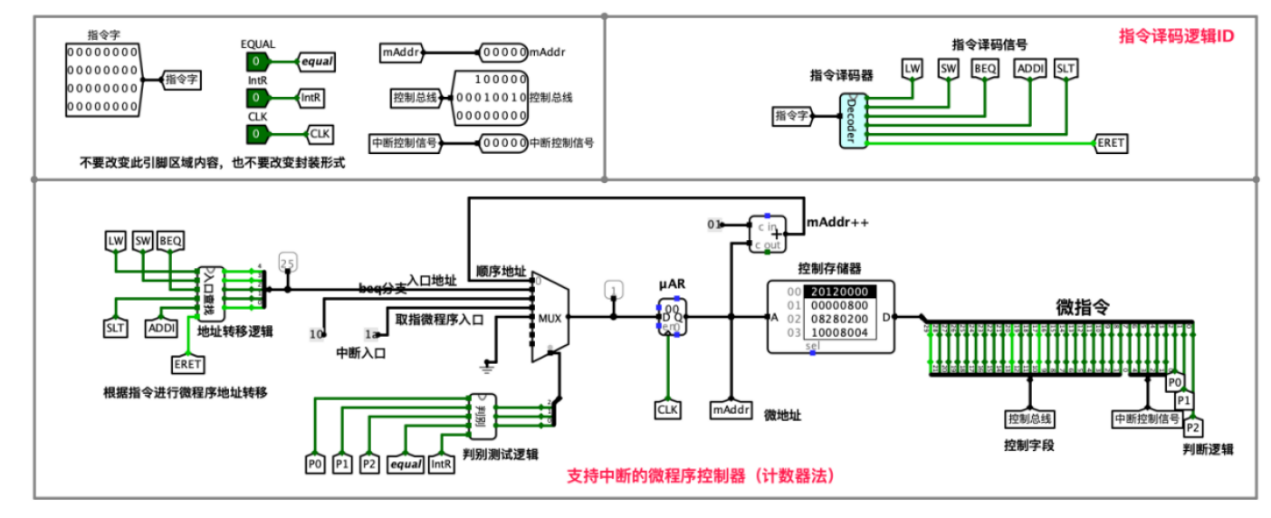
* ****微程序控制器

图1.16单总线结构现代时序（带中断）微程序控制器总体结构图

实现逻辑：其他功能模块已经在之前说明，这里最重要的就是下一条微程序的地址：需要根据微程序条件判断逻辑的结果来进行多路选择

* 硬币线时序产生器

实现逻辑：用Excel构造组合逻辑，然后在logism生成电路。

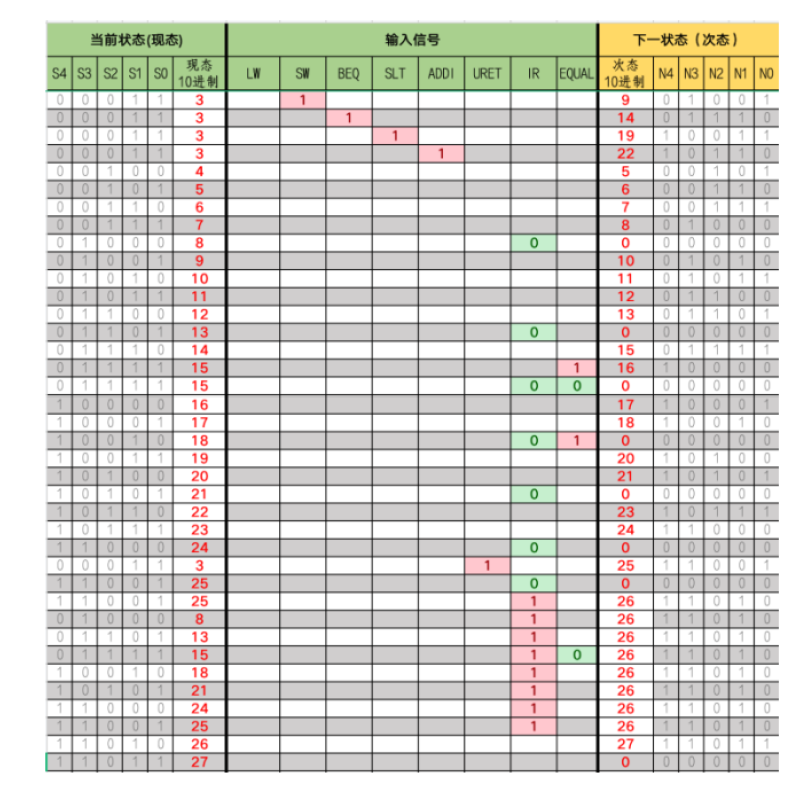


图1.17单总线结构现代时序（带中断）硬布线时序产生器组合逻辑设计表

* 硬布线控制器

输入：指令字，EQUAL和时钟

输出：控制总线输出。

实现逻辑：主要就是状态寄存器右边部分是当前状态，左边部分是次态，将当前状态和指令译码器的输出作为FSM的输入连线即可。因为硬布线的状态和微程序的地址是一一对应的，这个时候可以借用之前做的微程序控制寄存器来读取控制总线输出。

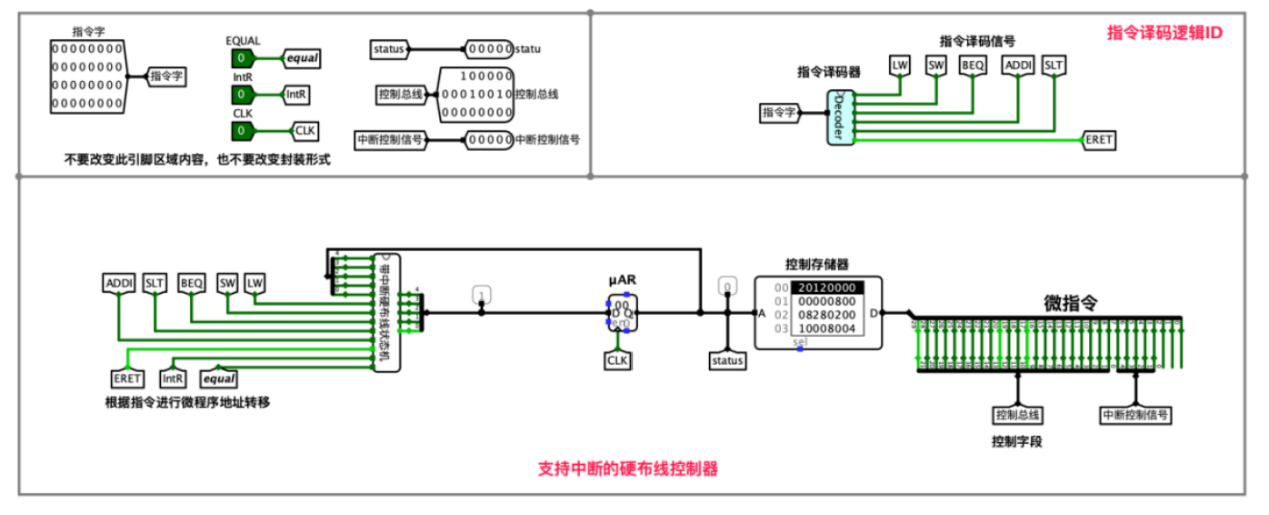
****

图 1.18单总线结构现代时序硬布线控制器（带中断）总体结构

* 中断控制器件

IE寄存器：关中断的时候异步置为1，开中断的时候异步置为0，输出为寄存器保存内容相反的部分。

mEPC寄存器：和数据通路的其他锁存器是一种构造，就是在写入问好为0的时候，通过使能端控制寄存器忽略时钟信号，不把输入端内容寄存下来，锁存器输出接三态门，三态门只有在EPC输出信号有效的情况下，才能进行输出，不能进行输出的时候三态门阻挡。

中断地址查找：根据中断控制的中断类型输出进行分支判断，分支判断使用多路选择器，对于输出类型为0&3的就不与处理，对于分支1，接上代表1号中断的中断入口地址的常量，对于分支2接上对于2号中断的中断入口地址常量。

对于中断地址的访问，首先查看这个不带中断的程序，发现程序在00000063停止，第42行是正常程序的最后一条；对于带中断的程序，从主程序在第42行结束，从第43行开始就是中断程序，接着找到中断程序的第一条指令00810113，在第43行，我们接着往下找，发现在第61行也发现了指令00810113，推测这是第二个中断程序的第一条指令，那么中断入口指令分别是第41条指令和第59条指令；又已知一条指令占4个字节，所以说我们可以知道第一个中断指令的地址就是[41\*4]10和[59\*4]10。我们可以使用Rars来分析汇编语言中各个标签的地址,也可以得到答案。

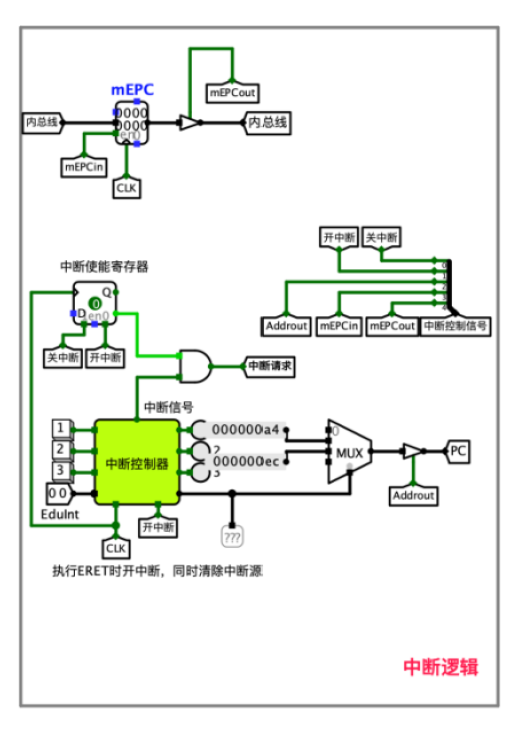


图1.19总线结构现代时序（带中断）中断逻辑总体结构

### 2.3单总线结构三级时序变长指令CPU

* 指令译码器
* 硬布线时序发生器有限状态机

实现逻辑：利用Excel表构造组合逻辑，然后在logism生成电路



图 1.20单总线结构三级时序硬布线时序发生器FSM组合逻辑设计表

* 硬布线时序发生器组合逻辑输出组件

实现逻辑：用Excel表构造组合逻辑，然后在logism生成电路

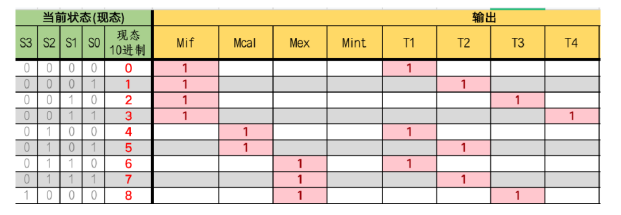
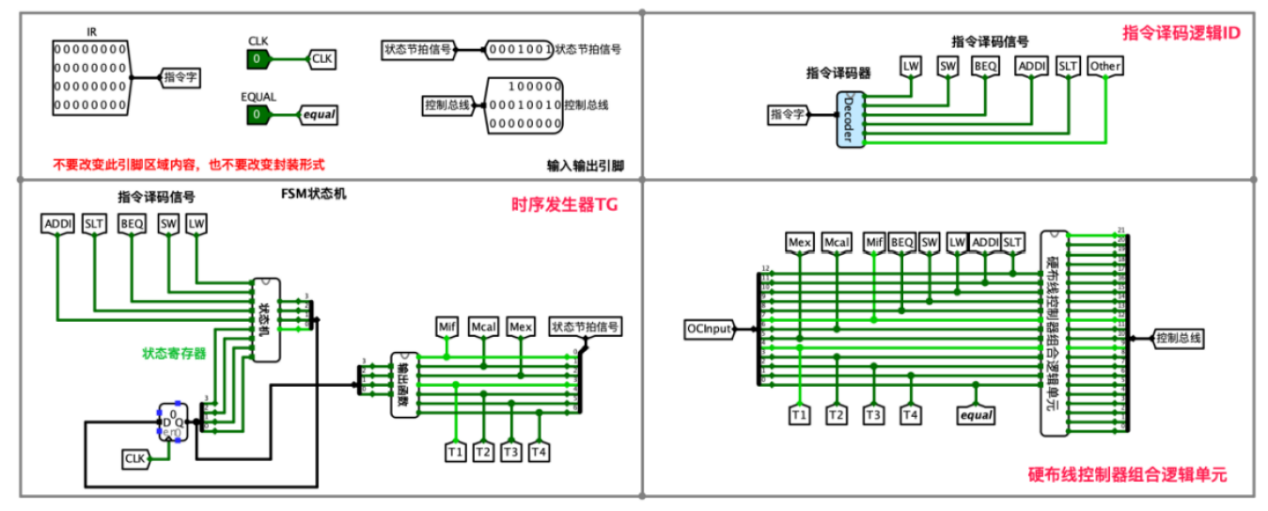


图 1.21单总线结构三级时序硬布线时序发生器输出函数逻辑设计表

* 硬布线组合逻辑输出组件

实现逻辑：用Excel表构造组合逻辑，然后在logism生成电路。



硬布线控制器****

实现逻辑：主要的就是状态寄存器右边部分是当前状态，左边部分是次态，将当前状态和指令译码器的输出作为FSM的输入连线即可。根据当前状态序号，可以通过硬布线组合逻辑输出器件输出各种控制信号的输出。

图1.22单总线结构三级时序硬布线组合逻辑输出函数逻辑设计表