

计算机组成原理实验二实验报告

**学 院： 计 算 机 工 程 学 院**

**班 级： 计算2114**

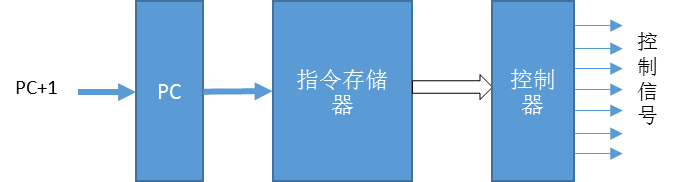
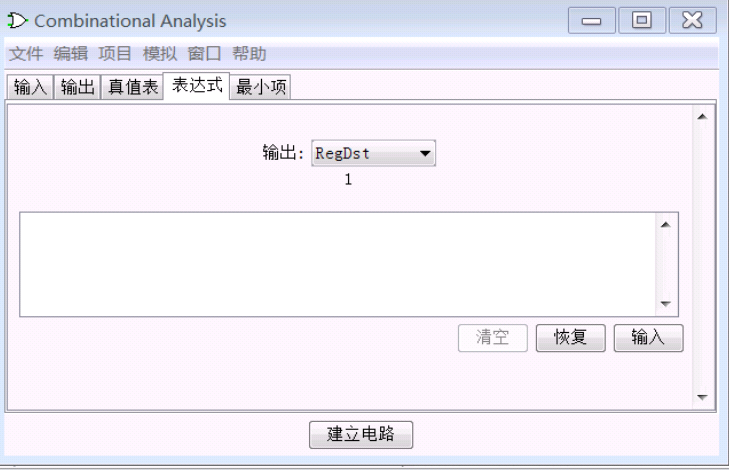
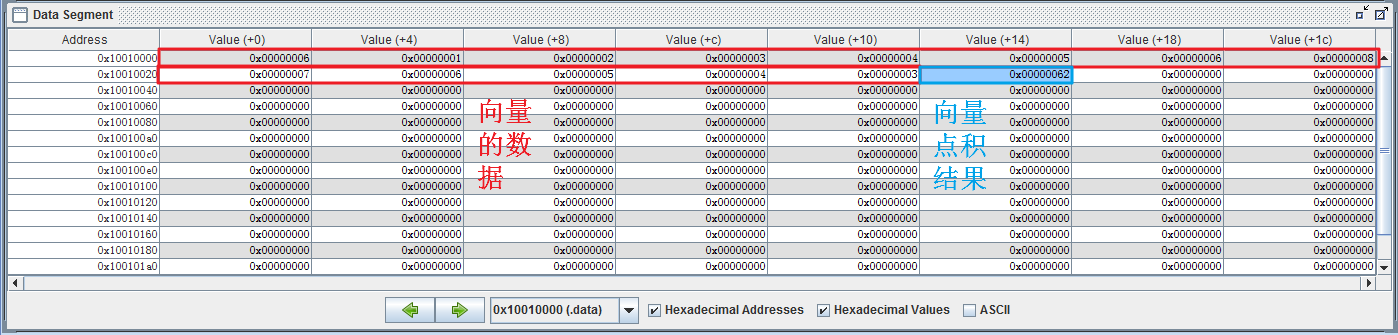
**姓 名: 庄佳强 章立早**

**学 号： 202121331104 202121331118**

二〇二二 年 10 月24 日

# 实验二.F/G

# 一 题目要求

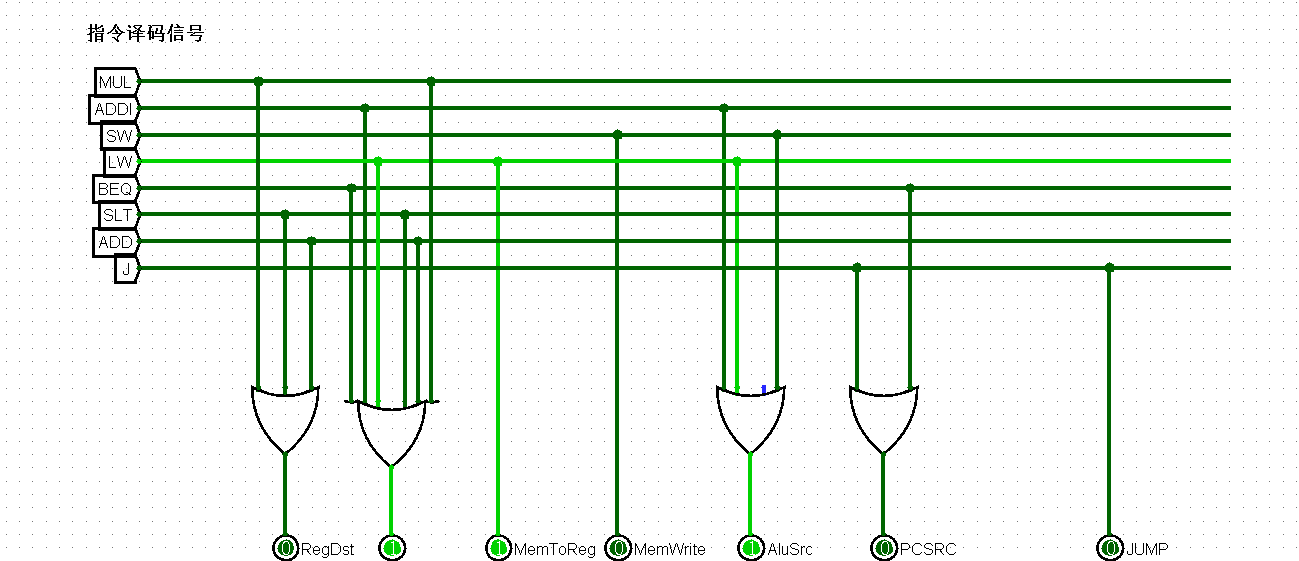
* 1. 设计你的MIPS处理器  
      当你把测试程序的代码段加载到Logisim的存储器当中，剩下的任务就是利用你所学的知识，围绕指令存储器去设计相应的电路，令你的电路可以完成“取出指令、译码、运算、访问数据存储器和写回寄存器文件”这些关键功能。你可以充分利用Logisim提供的电路模块、预先设计好的的ALU和寄存器文件，构建出你自己的MIPS处理器。  
      在设计的过程中，你可能会遇到的难点包括：
  2. 一次性完成9条指令的设计后再进行处理器调试，可能会是一个比较复杂的过程，建议你采用增量设计的思路，添加一条指令后就测试一条指令。
  3. Logisim当中的ROM的地址线宽度有限，因此实验提供的参考设计中只取用PC的[13:2]的部分，用于控制指令存储器的地址。由于我们的测试程序会访问到的地址空间很小，这种做法并不会影响测试程序的运行。
  4. 为了能够控制你设计的ALU和寄存器文件的运行，你需要设计一个电路，根据指令产生相应的控制信号，这个原理如下图所示：  
       
     然而，如果要根据指令和控制信号的真值表关系去手工画出控制器的电路，将是一个相当繁琐的过程。你可以利用Logisim提供的Combinational Analysis功能，手工输入逻辑表达式，让Logisim帮你画出控制器的电路。这个功能如下图所示：  
       
     请自行查阅Logisim的帮助手册，掌握这个功能的用法。（当然，你也可以不用这个功能，利用与/或/非等逻辑门实现控制器电路。）
  5. 当你完成设计后，你可以利用Logisim提供的“时钟前进一步”功能进行单步调试，它对应的快捷键是Ctrl-T。使用单步调试将允许你仔细检查你的处理器的各条线路的状态。
  6. 测试你的处理器  
      当你设计完你的处理器后，测试程序的运行结果必须与MARS仿真器的运行结果一致。MARS仿真器的运行结果如下图所示：  
     你可以打开数据存储器观察其保存的计算结果是否正确。  
      在实验提供的汇编程序的最后一行，有一条空指令nop。你需要在你设计的处理器中检测是否出现了空指令，当遇到空指令时，停止系统的运行，并点亮一个LED指示灯。在你的实验报告中展示：（1）你在Logisim当中添加新指令的过程；（2）你是如何验证计算结果的正确性的。

# 二 设计思路

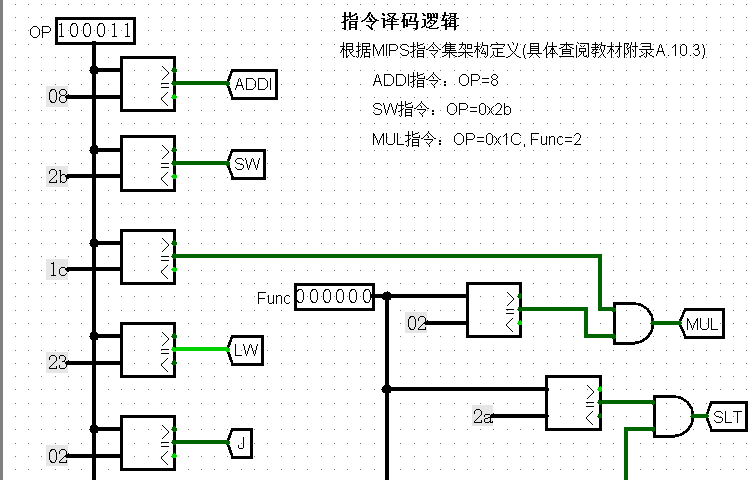
1. 通过实验之前的个个热身和演示，大致掌握了设计CPU的方法，在设计中，只想要完成控制器中剩下的指令op所对应的译码信号的解译,利用MARS仿真器当中的MIPS-XRAY功能查看在测试程序执行过程中处理器的工作方式。方便连线。
2. 在main中，要完成beq的跳转和j的跳转，用Pcsrc 作为一个信号，而当x==y即Equ时进行beq跳转，为了方便，为j指令单独设立了一个jump的译码信号，当j 指令时，jump会辅助 进行j 跳转。
3. 之后就可以导入数据进行测试。

# 三 成果展现

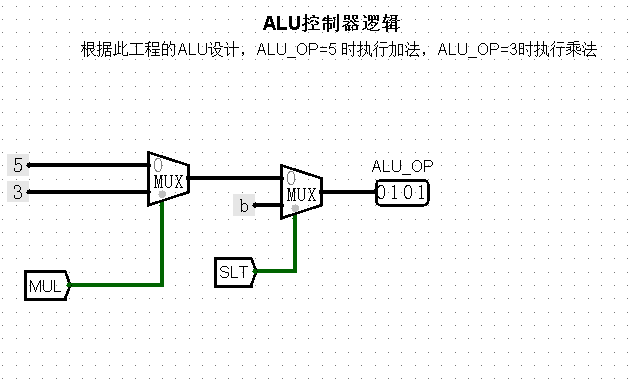
### 译码信号部分：



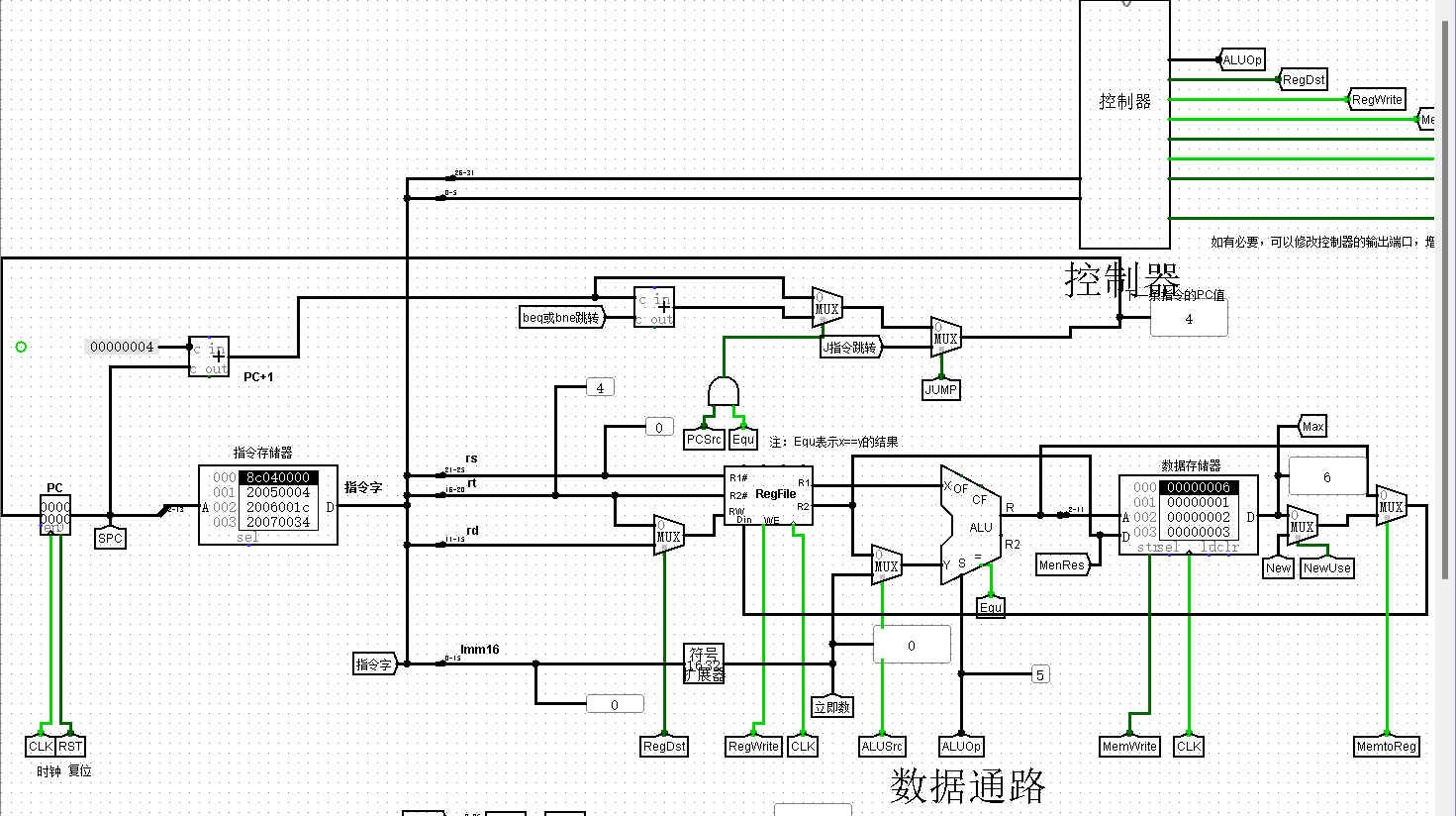
### Op部分：



### ALU控制：



### Main部分：



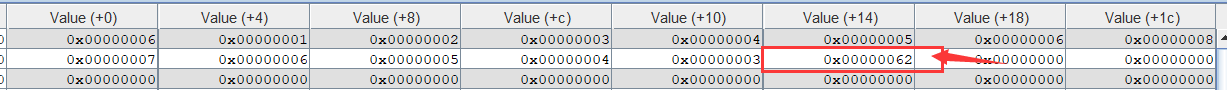
### 特殊部分：



一个用于beq跳转，一个用于运算j跳转指令

### 计算结果：





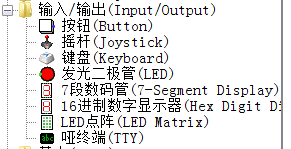
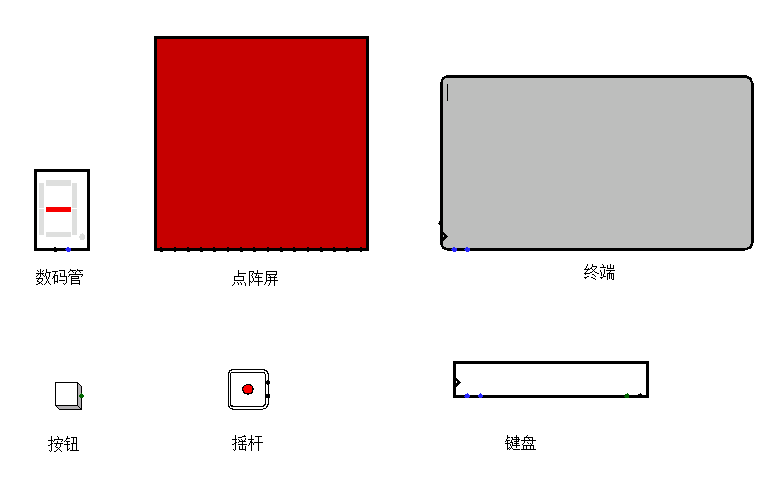
具体放在视频演示部分。

# 四 实验细节

1. j前要注意先减掉3000 因为这是 指令的初始值。
2. Beq的立即数只是跳转的指令数，在输入到pc之前还要先乘4。

# 实验.H

# 一 题目要求

生活中需要创新和创意，它可以让我们的生活变得更美好。在这个实验中，你可以试着进行一些“微创新”，只需要关注一些大多数人不容易注意到的细节，你就有可能让你的设计变得与众不同。Logisim为你提供了丰富的输入和输出设备：  
  
  
选择一个你感兴趣的应用，这个应用能够使用到Logisim提供的几种输入外设之一（按钮/摇杆/键盘）和几种输出外设之一（数码管/点阵屏/终端）。你的硬件系统将以自己设计的MIPS处理器为核心，通过内存映射的方式挂载外设；你的软件系统将以MIPS汇编程序为核心，通过SW/LW指令访问外设，并通过其他算术指令完成数据的计算、控制流等。

# 二 设计思路

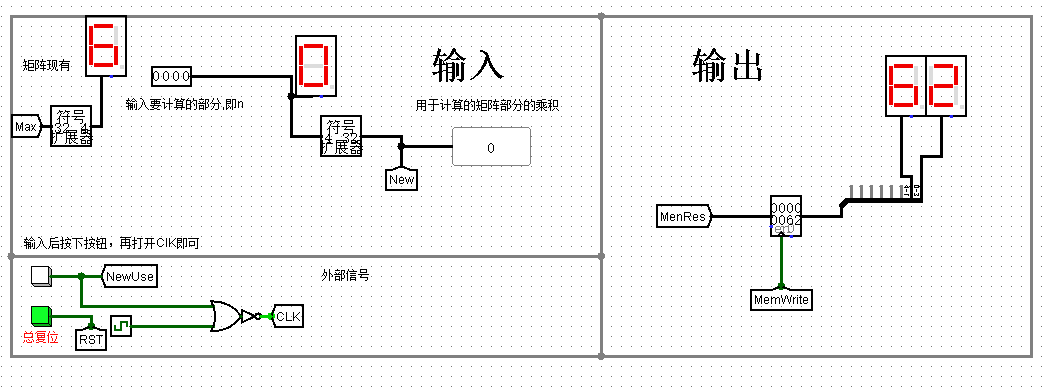
有时候我们其实不需要计算整个矩阵的乘积，我们只需要获取n矩阵的乘积，所有我设计了一个输入和输出来方便我们进行n的操作，和利用了数码管使得结果可视化。

设计一个简单，输入和按钮操作。当按下输入要计算的n后，按下按钮就可以实现局部矩阵的计算。

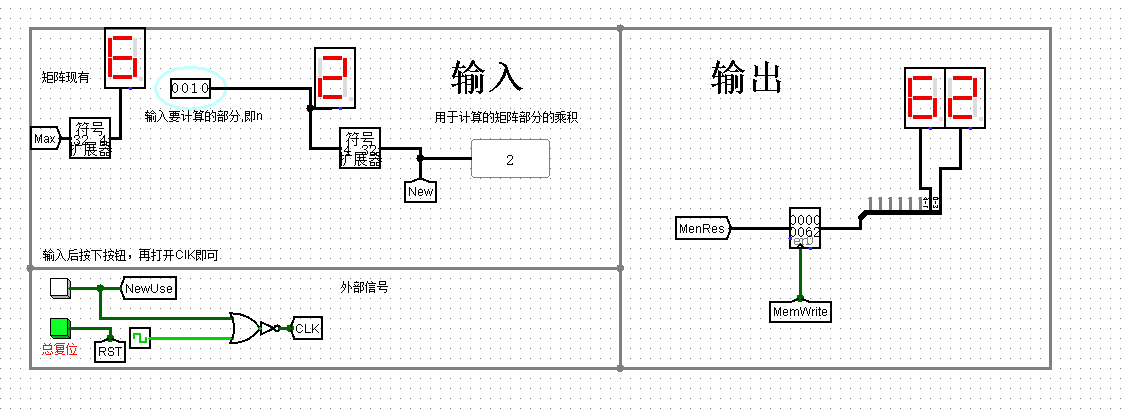
当要输出结果时，使用数码管输出即可。

# 三 成果展现

### 设计部分：

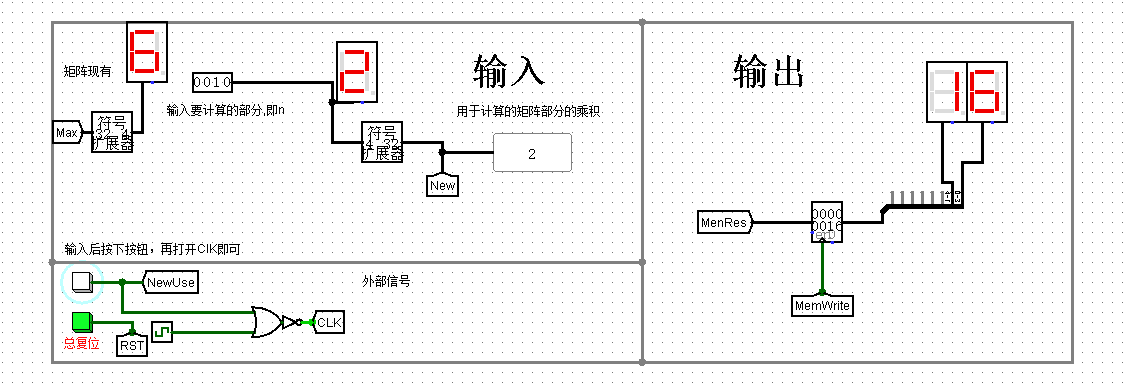


### 初始化：



接着就可以开始运算。。

### 结果：



# 四 实验细节

1. 如果设置n和CLK同步时会导致设置失败，所以使用非门使他们的运算分开。
2. 当MemWrite时表示 要读入数据，这时候就可以利用这个同步把数据读入到寄存器中，输出到数码管中。

# 实验.附加

# 一 题目要求

我们在课堂上展示过一个完全由Verilog描述的迷你MIPS处理器设计，然后将其烧写至FPGA芯片后，运行MIPS汇编程序并将循环递增计数的值显示在7段数码管上。在这个任务当中，你的任务是修改设计（方法不限，但必须保留MIPS汇编程序和MIPS处理器的基本形式），将循环递增计数改变为循环递减计数。当你完成之后。

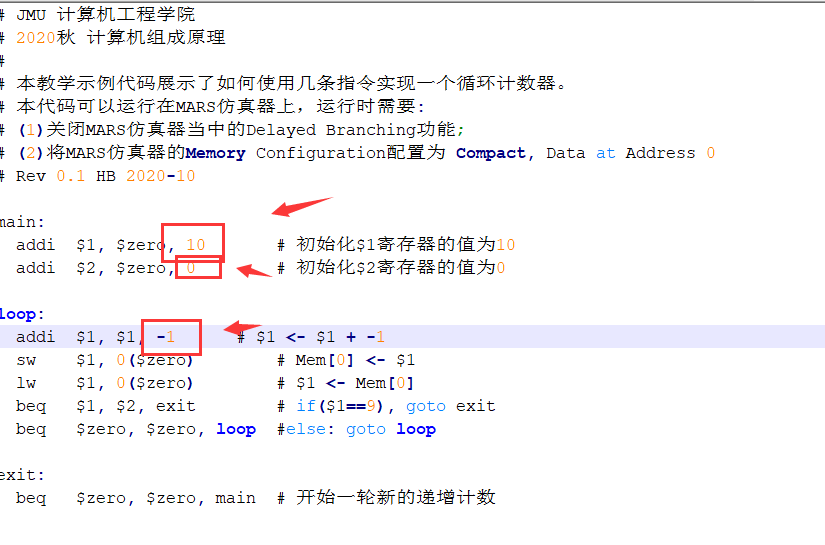
# 二 设计思路

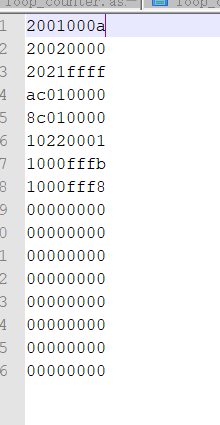
我是从软件的角度去思考问题，因为基础的部分都设计好，通过考虑，通过修改代码时最简单的解法。

把$1修改成10，$2修改为0 ADDi 加1变成减1.就完成了递减的操作。

最后修改一下编码就好了。

# 三 成果展现





演示在视频中。

# 四 实验细节

$1 修改成10而不是9原因是：程序是在sw后再读入到FPGA中，想让FPGA显示出9就应该先减-1后sw是9可以。

# 实验总结

1. 一开始是手足无措的，但在反复观看实验指导书和MARS仿真器当中的MIPS-XRAY后逐渐有了灵感，卡卡停停的设计出了个个指令的译码信号。完成的最难的一步。
2. 后面就是j指令的跳转是一个难题，为此最后使用了一个技巧跳转了j指令执行。
3. 通过这个小实验，学到了流水线的基础开发，实践把指令码实现在屏幕中，从字符变成了行为。