**计算机组成原理实验报告**

数据通路图：



指令系统设计：





实验过程：

1. 本次实验耗时较长，我们小组是先设计了数据通路图，再到指令系统的设计，最后用高级语言实现了从1+2+3+….+100。程序可以进行单步调试、可以修改当前状态、以及让它顺序执行。
2. 汇编代码说明：

Start//表示程序的开始

movl $0, R1 //R1用于存放运算结果，初始值为0

movl $1, R2 //R2用于记录下一个需要加的数

jmp .L2 //转跳到L2

.L3:

movl R2, R0 //R0存放的是当前要加的值

addl R0, R1 //让R0+R1结果放到R1

addl $1, R2 //下一个加数+1

.L2:

cmpl $100, R2 //比较下一个要加的数是否大于100

jle .L3 //如果R2 <= 100 则转跳到L3

movl $0, R0 // R2 > 100则调出循环

end

1. 输出说明：

初始化时PC = 1,其他参数如图所示

PC：程序计数器

AR：地址寄存器

Z：暂存器

D：缓冲寄存器

IB:内部总线

IR:指令寄存器

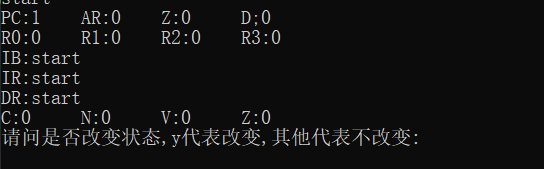
DR:数据缓冲寄存器

状态寄存器：

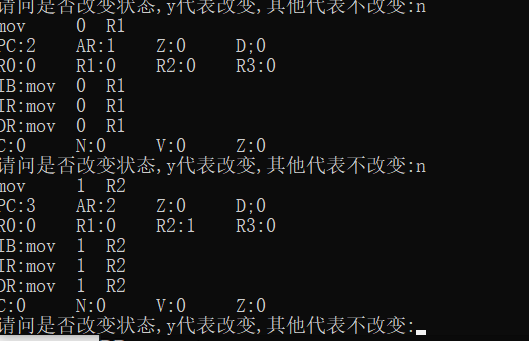
N  N=1表示运算的结果为负数，N=0表示运算的结果为正数或零.  
Z  Z=1表示运算的结果为零，Z=0表示运算的结果非零。

C 当运算结果产生了进位时（无符号数溢出），C=1，否则C=0。

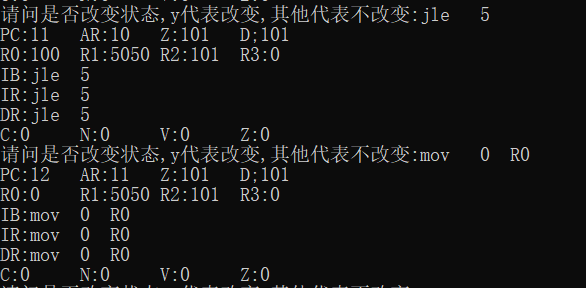
V V=1表示符号位溢出



输入n，即不需要临时改变内部参数的值，则按顺序执行执行



执行结果如下，当R2 = 101时跳出循环，得到结果R1 = 5050, 再把R0的值设为0



实验体会：

软件/硬件实现的区别：

1. 软件最小操作单位是字节，而硬件的最小操作单位是比特。
2. 在硬件实现的方式中需要对每一个器件进行详细的定义，比如输入输出的接口等等。而软件对这些实现可以根据实际情况随意改变。
3. 硬件实现的方式实现并行执行，软件实现是顺序执行。所以在硬件实现时，完成一个操作时，可以将多个控制信号同时赋值。在软件实现时需要每改变一个信号就要让信号执行完成后再对下一个信号做出改变
4. 在硬件实现的过程中是将硬件内部结构连接起来，所以不可避免地会出现一些差错，而对于软件实现则会稳定一些。

收获和心得：

1. 在此之前认为设计CPU遥不可及，因为CPU的内容太多，自己也不清楚CPU是如何运行的。但经过课程上的学习和老师对CPU的指令运行过程的讲解，让我对CPU的实现有了比较清晰的想法，先弄懂指令执行的过程，过程中经过的器件，明白他们的功能，然后按自己的理解将数据通路图画出来，这样对CPU整个结构和运行的逻辑大概有了了解。
2. 在设计数据通路和指令系统的时候对CPU内部各器件的作用结构等有了更深的了解而且之后对指令的执行过程以及它的格式，分类有了清楚的认识。
3. 在编写程序实现时，指令通过译码发出控制信号，但是手动操作控制信号实在太繁杂。这样我联想到课上提到的微程序控制器实现的优势。将课上所学与实践相联系，感觉收获颇丰。