## 实验6：CPU设计综合

### 一、实验目的

1. 掌握不同指令数据通路的实现方式。
2. 掌握CPU基本结构，并学习不同部件级联调试方法。
3. 掌握RISC-V汇编程序设计，并转换成机器代码的方法。

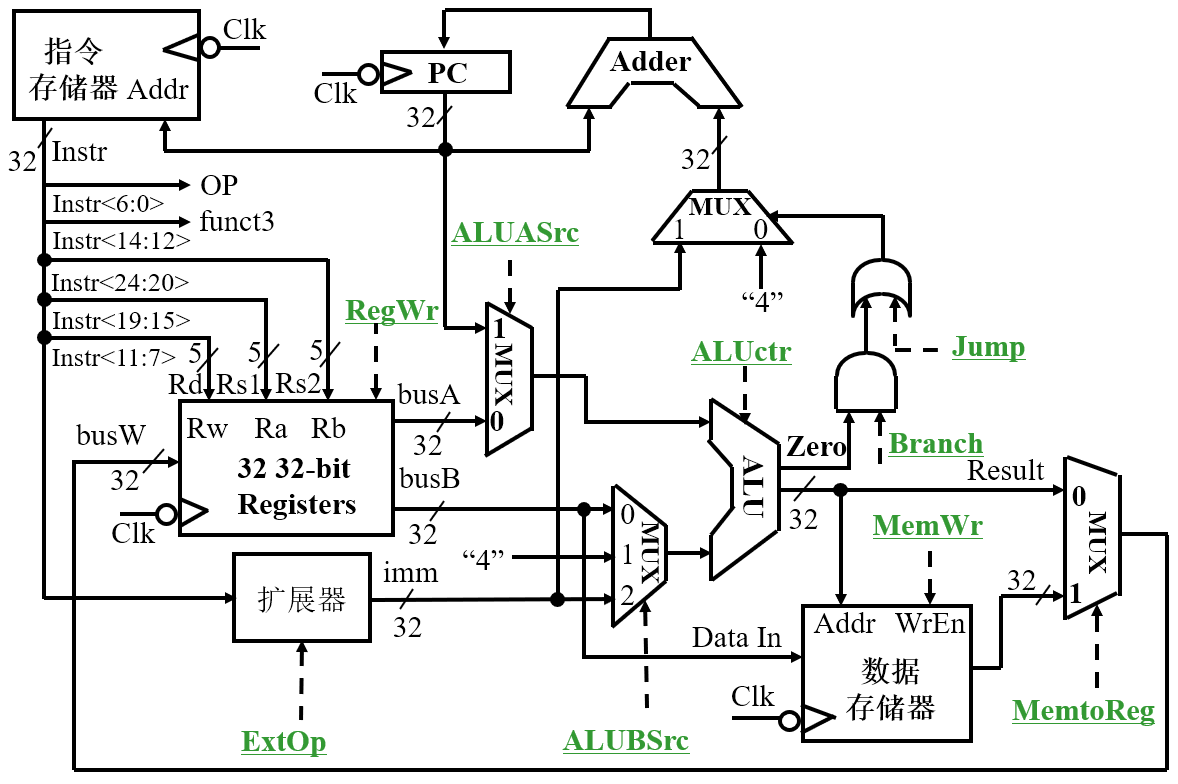
### 二、实验环境

Logisim-ITA V2.16.1.0。

RARS：RISC-V模拟器工具<https://github.com/thethirdone/rars>

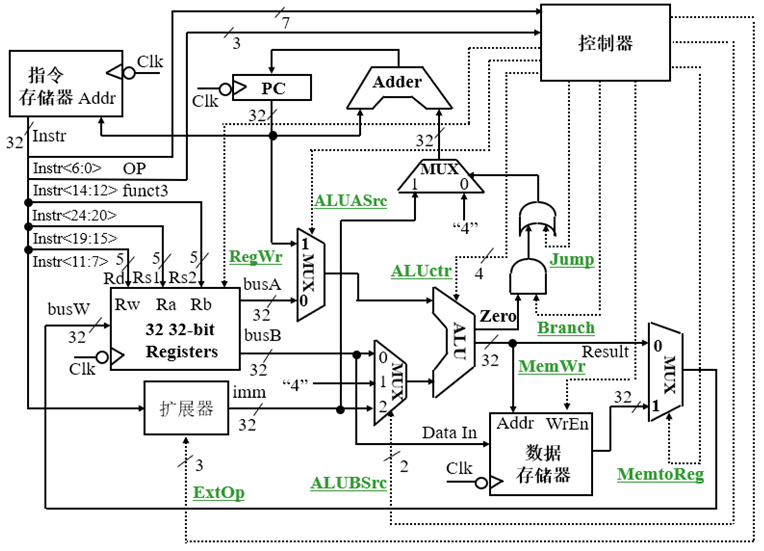
### 三、实验内容

1. 根据给出的9条RISC-V指令的数据通路原理图，实验并验证数据通路的功能。



数据通路原理图

1. 根据给出的单周期RISC-V CPU结构原理图，实现并验证CPU的功能。



单周期RISC-V原理图

1. 编写一个仅使用已实现的9条RISC-V指令的汇编程序，输入参数和输出结果保存在RAM中，为了便于调试，建议增加复位按钮，用于复位寄存器中数据。根据指令格式转换为机器代码表示，写入到指令寄存器中，并启动程序执行，写出执行每个步骤后寄存器中保存的值，验证执行结果的正确性。

指令编码格式

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 指 令 | funct7(31-25) | rs2(24-20) | rs1(19-15) | funct3(14-12) | rd(11-7) | op(6-0) |
| add rd, rs1, rs2 | 0000 000 | rs2 | rs1 | 000 | rd | 011 0011 |
| slt rd, rs1, rs2 | 0000 000 | rs2 | rs1 | 010 | rd | 011 0011 |
| sltu rd, rs1, rs2 | 0000 000 | rs2 | rs1 | 011 | rd | 011 0011 |
| ori rt, rs1, imm12 | imm{11:0} | | rs1 | 110 | rd | 001 0011 |
| lui rd, imm20 | imm[31:12] | | | | rd | 011 0111 |
| lw rd, rs1, imm12 | imm[11:0] | | rs1 | 010 | rd | 000 0011 |
| sw rs1, rs2, imm12 | imm[11:5] | rs2 | rs1 | 010 | imm[4:0] | 010 0011 |
| beq rs1, rs2, imm12 | imm[12][10:5] | rs2 | rs1 | 000 | imm[4:1][11] | 110 0011 |
| jal rd, imm20 | imm[20][10:1][11[19:12] | | | | rd | 110 1111 |

示例程序：编写一个计算1+2+…+n的累加和程序，从内存0x00000中读入参数n，通过循环累加的算法计算结果，并保存到到内存地址0x00001。

1）、先编写好RISC-V汇编程序，然后读入到RARS中，编译调试，通过仿真执行后，把汇编语句转换成机器代码，写入到一个机器代码文件中。

2）、在Logisim中，打开CPU程序，用鼠标右键点击IFU模块，选择 View IFU；在IFU的RAM模块上，用鼠标右键点击，选择load image，装入机器代码文件。

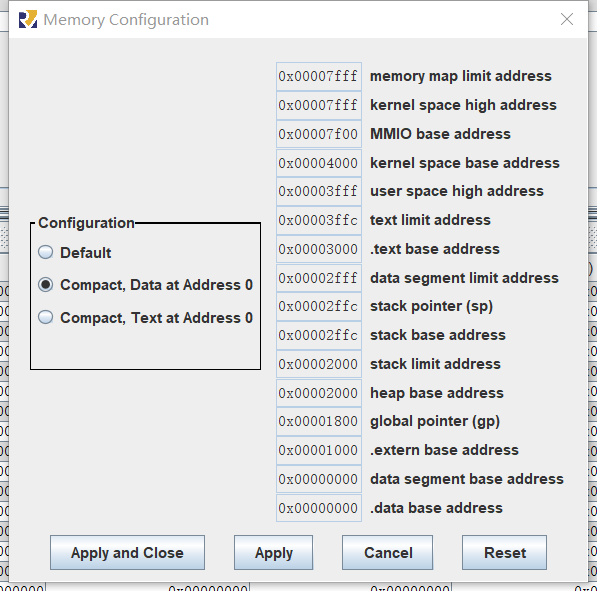
3）、在CPU模块中，选中RAM，用鼠标右键点击，选择Edit Content，在初始0x0000位置处，写入参数值n（如64）

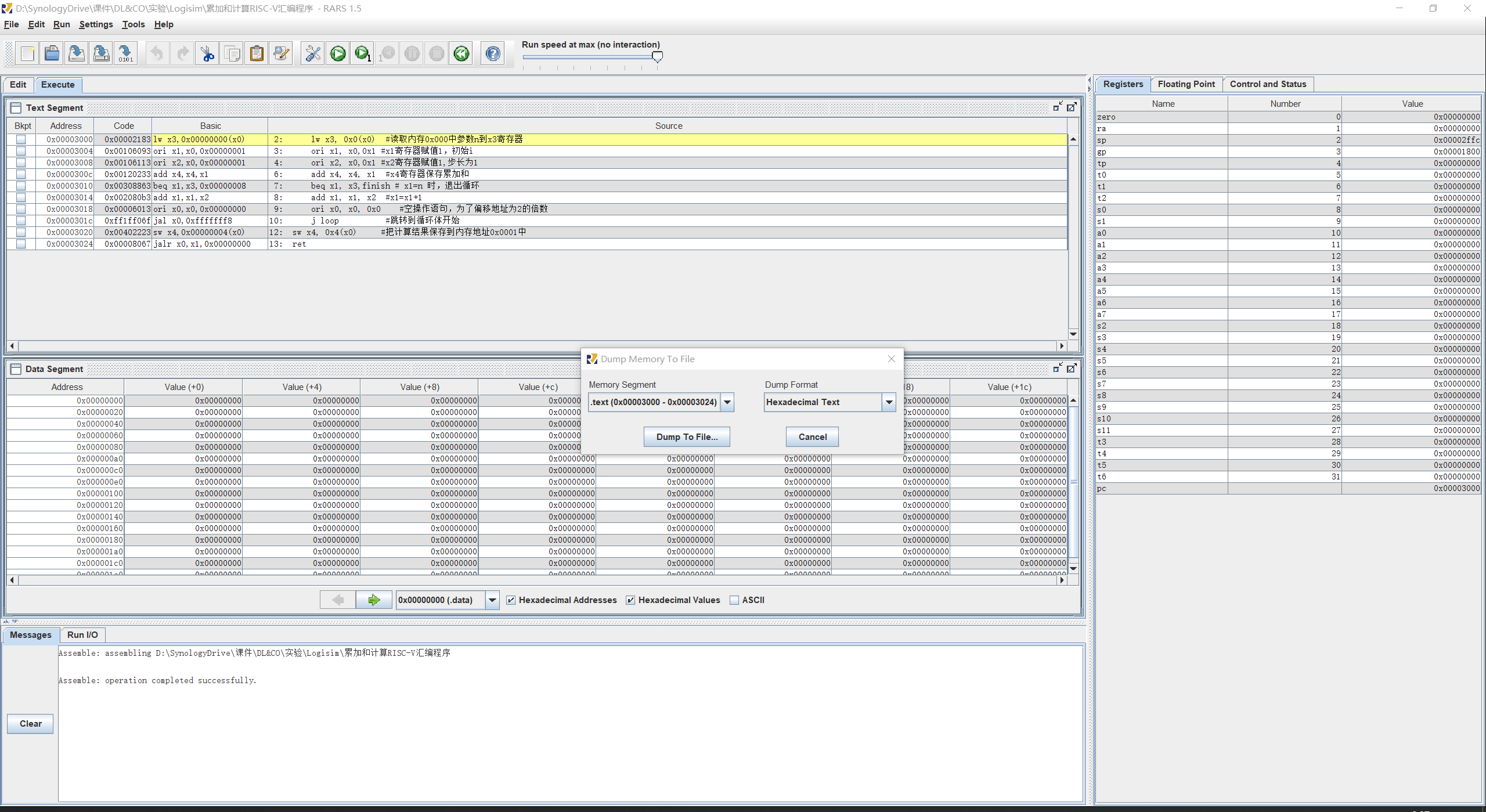
4）、在Logisim的仿真菜单下，选择合适的频率，选中Ticks Enable，则开始自动执行机器代码，程序执行结束后查看内存0x0001处的数据结果。

**RARS模拟器使用提示：**

1、虚拟存储模式设计：由于单周期CPU设计中没有考虑内存管理单元，指令存储器和数据存储器分离设计，运行时采用物理地址，并且起始地址都是0。为了使编程的测试程序能在RARS中仿真运行，需配置RARS模拟器中的RISC-V虚拟存储模式，在菜单Setting中的Memory Configuration选项中设置为Compact，Data at Address 0。这样数据段的起始地址就从0开始。

2、机器代码导出：在RARS的File菜单中的Dump Memeory To File功能将汇编程序段的机器指令和数据段的数据导出，选择导出类型为16进制文本格式，导出到某个文本文件，然后插入首行内容为：V2.0 raw，则导出文件可以在Logisim中选择Load Image直接加载到ROM或RAM组件中。





lw x3, 0x0(x0) #读取x0寄存器里数值加上0x0的内存地址（0x0000）中的数值到x3寄存器。

ori x1, x0,0x1 #x1寄存器赋值1，初始i

ori x2, x0,0x1 #x2寄存器赋值1,步长为1

loop:

add x4, x4, x1 #x4寄存器保存累加和

beq x1, x3,finish # x1=n 跳转

add x1, x1, x2 #x1=x1+1

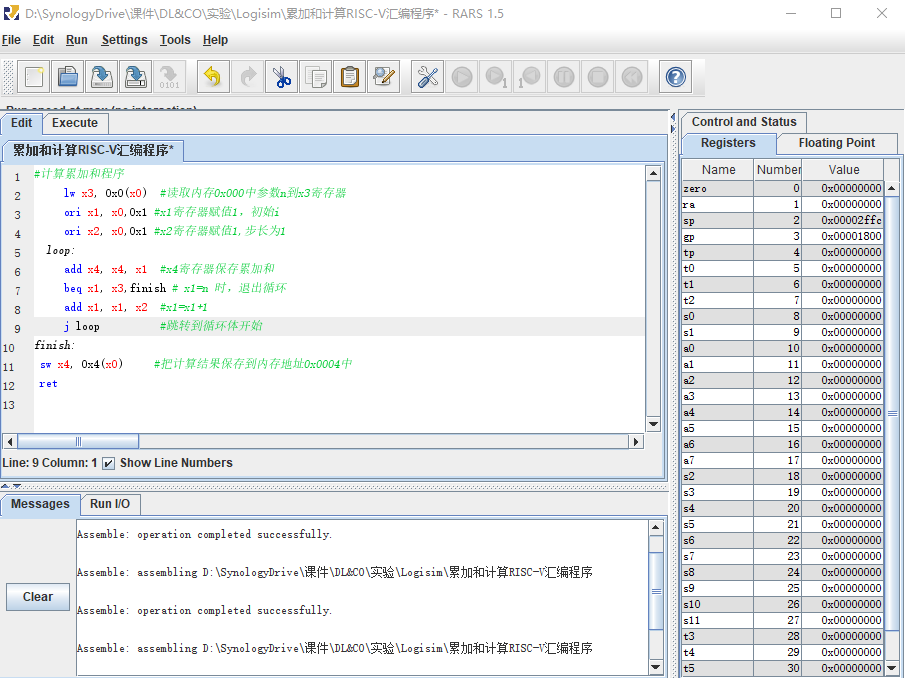
jal x0, loop

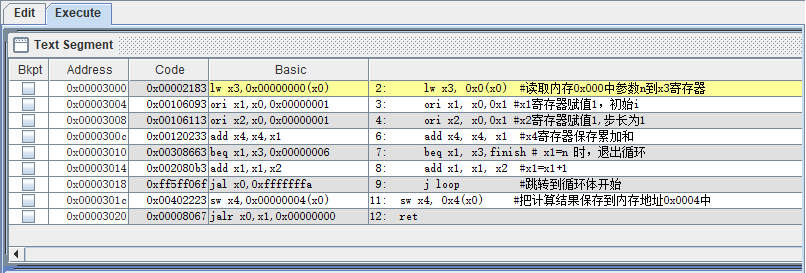
finish:

sw x4, 0x4(x0)

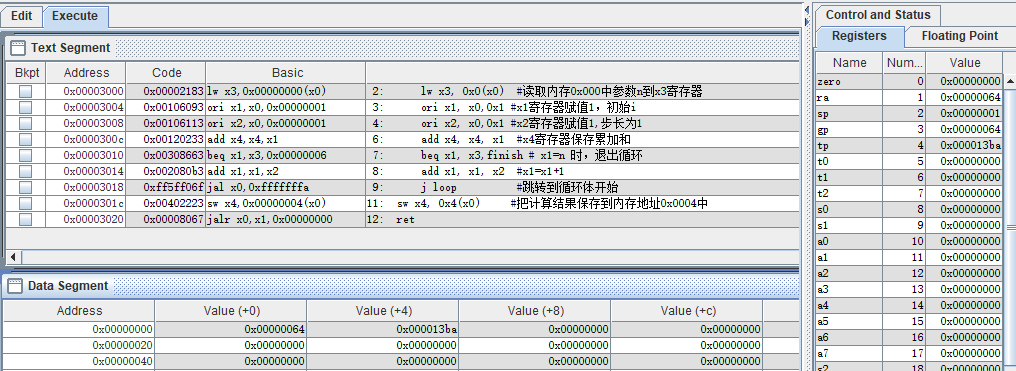
jal x0,finish

在RARS中编写/读取汇编程序，通过编译后，执行Aseemble（F3）命令，生成机器代码。设置虚拟存储初始地址为0，进行仿真运行。





在内存地址0x0000处设置参数为64，点击单步执行，得到每一条指令执行后的寄存器中的数值，执行结束后，得到最终结果。

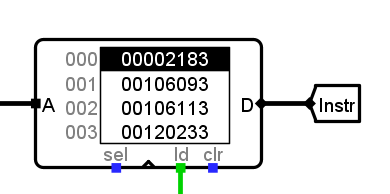


导出机器代码到文件test.o：

v2.0 raw

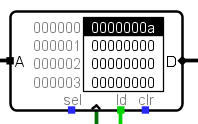
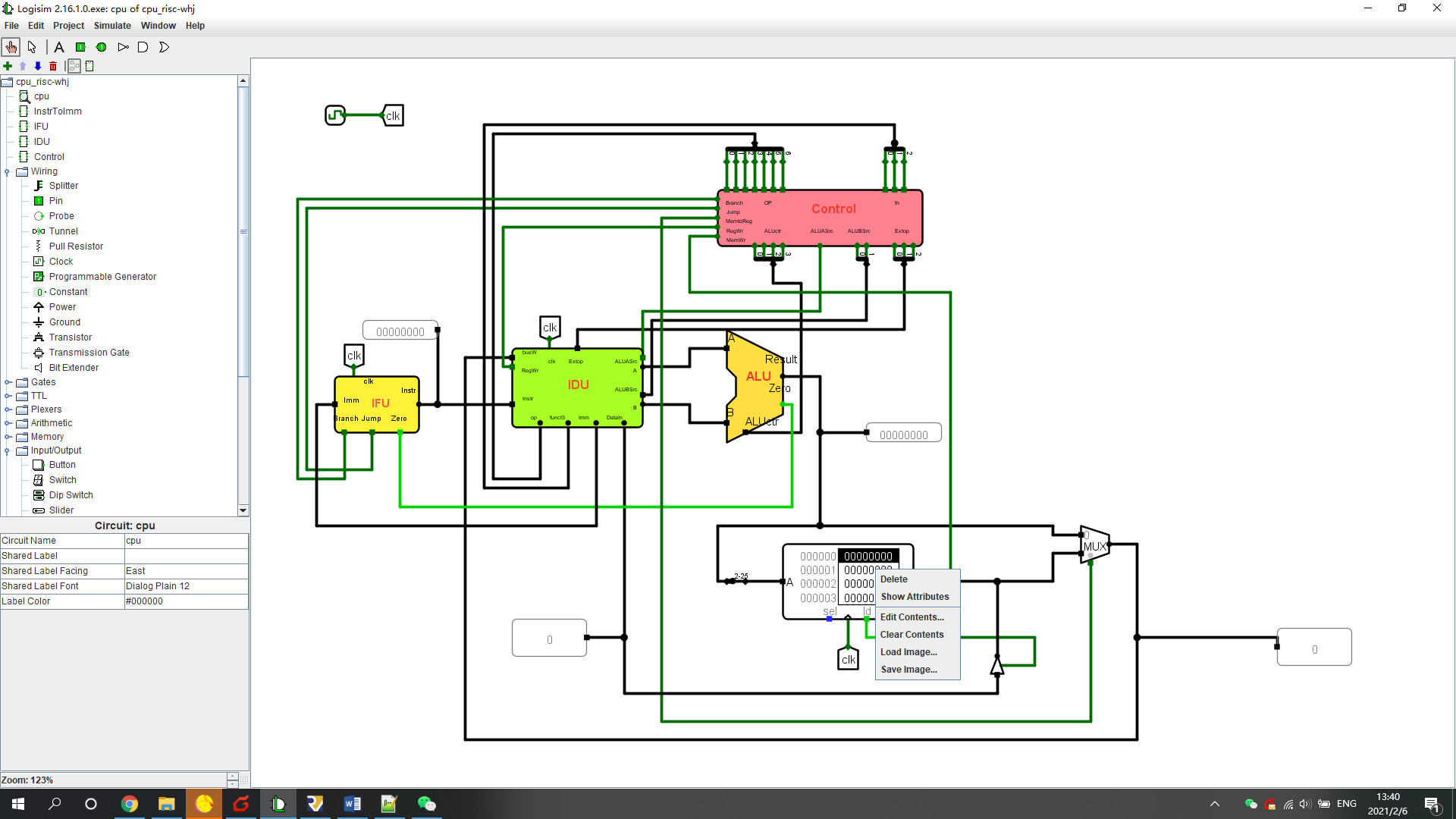
00002183 00106093 00106113 00120233 00308663 002080b3 ff5ff06f 00402223 00008067

在CPU主程序中，选中IFU模块，用鼠标右键点击，选择View IFU，进入IFU子程序，选中指令寄存器，用鼠标右键点击，选择Lord Image 装载机器代码文件。

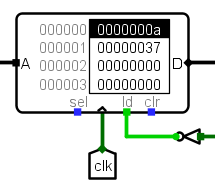


指令寄存器

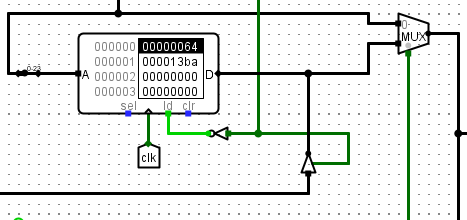
在CPU主程序，选中内存模块，用鼠标右键点击，选择Edit Contents，在初始地址0x0处输入初始参数n，如0x0a。



选择合适的时钟频率，选中Ticks Enable （Ctril+k），执行程序结果为0x37。



修改输入参数为0x64得到结果为0x13ba。



### 四、思考题

1. RISC-V中0号寄存器的内容始终为0，则在寄存器堆的设计电路中需要做哪些修改？
2. 如果需要增加与运算和逻辑右移运算，分别需要修改电路中的哪些部分？
3. 编写一种排序算法汇编程序，并进行验证结果。

### 五、实验报告

1、 根据本次实验内容的要求，写出实验操作步骤，包括：电路原理图、功能表、仿真检测图、输入输出对应表、错误现象及原因分析、思考题等内容。以word或PDF格式提交

2、 将实验报告和电路图.circ文件打包上传到教学支撑平台的网站中。