计算机组成原理实验报告

一、 实验题目

利用 RISC-V 指令系统实现冒泡排序算法

二、 实验目的

基于 RV32IM 汇编，设计一个冒泡排序程序，并用 Ripes 工具进行调试执行，测量冒泡排序程序的执行时间

三、 实验平台

Ripes 工具

四、 实验过程

1. 设计思路
   临时变量 i 从 1 曾至数组长度 n-1，用两个寄存器分别从数组元素对应地址中读取 array[i-1]和 array[i]进行比较，若比较结果为逆序，则将两个元素交换。设置一个交换标志 flag，初始默认值为 0，若发生交换，将 flag 设置为 1。I 的值增加至 n-1 后，进行判断，若发生过交换，即 flag==1 则将 i 的值恢复为 1，flag 的值重新赋为 0，重新进行循环操作。
2. 代码解释
   排序算法的实现主要分为以下几个部分
   ▸输出 output：对数组元素进行输出，展示算法实现结果
   ▸外循环 outerloop：对临时变量，标志变量进行初始化
   ▸内循环 innerloop：取数组元素并比较，依据比较结果进入交换部分或者跳过交换部分
   ▸交换 swap：交换两个元素的值，并且值标志变量的值
   ▸跳过交换 skip-swap：对临时变量进行增加，依据运行结果判断是否继续内循环或外循环
3. 代码展示

```
1.  .data
2.
3.  n: .word 10
4.  array: .word 23,11,7,61,62,60,64,65,6,67
5.
6.  .text
7.
8.    lw a1 n                  #a1 = n
9.    la a3 array               #a3 -> array[0]
10.   li a4  0                 #a4 = i,now a4 = 0
11.   mv a5 a3                   #a5 -> array[0] now
12.
```

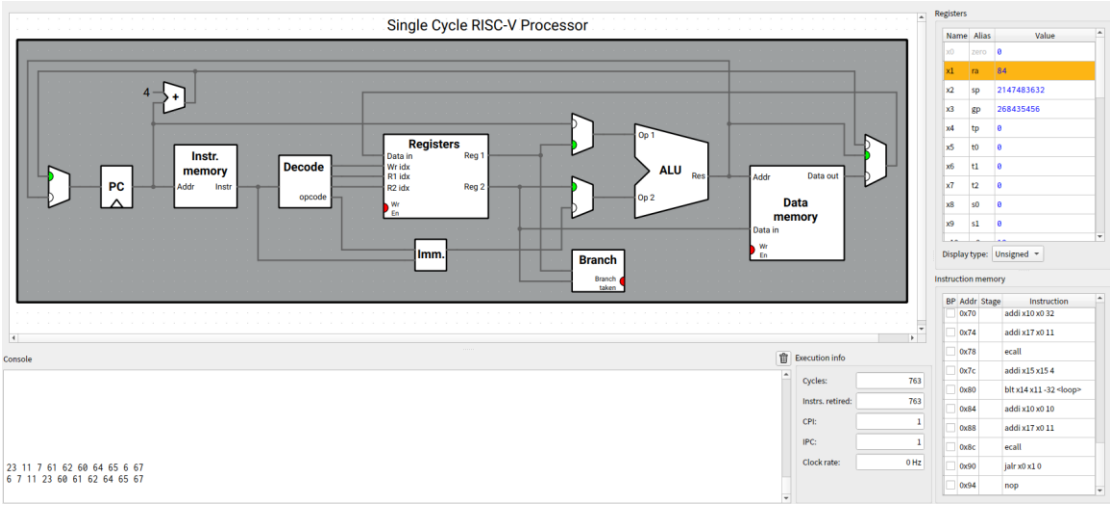
```
13.     jal x1 output               #Output array  first
14.
15.   outerloop:
16.     li a4 1                   #i = 1
17.     mv a5 a3                    #a5 -> array[0] now
18.     mv a6 zero                  #a6:swap flag: 1 else: 0,now flag = 0
19.
20.   innerloop:
21.     lw a2 0(a5)                 #a2 = array[i - 1]
22.     lw s2 4(a5)                 #a3 = array[i]
23.     bge s2 a2 skip_swap           #if array[i] >= array[i - 1] skip swap
24.
25.   swap:
26.     sw a2 4(a5)                 #array[i] = a2
27.     sw s2 0(a5)                 #array[i - 1] = a3
28.     addi a6 zero 1               #flag = 1
29.
30.   skip_swap:
31.     addi a5 a5 4                #a5 point to array[i - 1 + 1]
32.     addi a4 a4 1                #i = i + 1
33.     blt a4 a1 innerloop           #if(i < n) cotinue
34.     bne a6 zero outerloop         #if flag == 1 occurs continue outerloop
35.     jal x1 output               #output array after sort
36.     j Exit                    #Exit
37.
38.   output:
39.     li a4  0                  #a4 = i,now a4 = 0
40.     mv a5 a3                    #a5 -> array[0] now
41.   loop:
42.     addi a4 a4 1                #i = i + 1
43.     lw a0 0(a5)                 #a0 = a[i - 1]
44.     li a7 1
45.     ecall                     #output a[i - 1]
46.     li a0 32
47.     li a7 11
48.     ecall                     #output ' '
49.     addi a5 a5 4
50.     blt a4 a1 loop              #if(i < n) continue
51.     li a0 10
52.     li a7 11
53.     ecall                     #output '\n'
54.     ret
55.
56.   Exit:
```

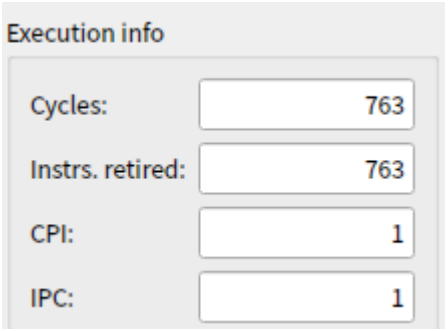
```
57.    nop                    #EXIT
```

五、 实验结果

▸测试数据 1：23 11 7 61 62 60 64 65 6 67

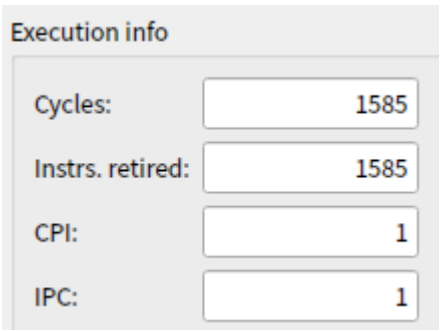
```
23 11 7 61 62 60 64 65 6 67
6 7 11 23 60 61 62 64 65 67
```





▸测试数据 2：0 0 0 78 65 43 44 5 9 1 1 -2 43 15 24 666

```
0 0 0 78 65 43 44 5 9 1 1 -2 43 15 24 666
-2 0 0 0 1 1 5 9 15 24 43 43 44 65 78 666
```



六、 总结反思

初次使用汇编语言实现已经学习过的算法，这个过程中我还是遇到了许多困难，例如对

指令没有熟练掌握，对汇编程序的结构还不是很熟悉，这次实验是一个学习的过程，也是一个克服定式思维的过程，我需要跳出以前编写 C 语言时的思维模式来思考问题，来实现算法。因此可以用收获颇丰来形容此次实验的过程。