计算机组成原理 实验报告

姓名：陈奕衡

学号：PB20000024

一、实验题目

- 实验三 汇编程序设计

二、实验目的

- 熟悉RISC-V汇编指令的格式
- 熟悉CPU仿真软件Ripes，理解汇编指令执行的基本原理（数据通路和控制器的协调工作过程）
- 熟悉汇编程序的基本结构，掌握简单汇编程序的设计
- 掌握汇编仿真软件RARS(RISC-V Assembler & Runtime Simulator)的使用方法，会用该软件进行汇编程序的仿真、调试以及生成CPU测试需要的指令和数据文件（COE）
- 理解CPU调试模块PDU的使用方法

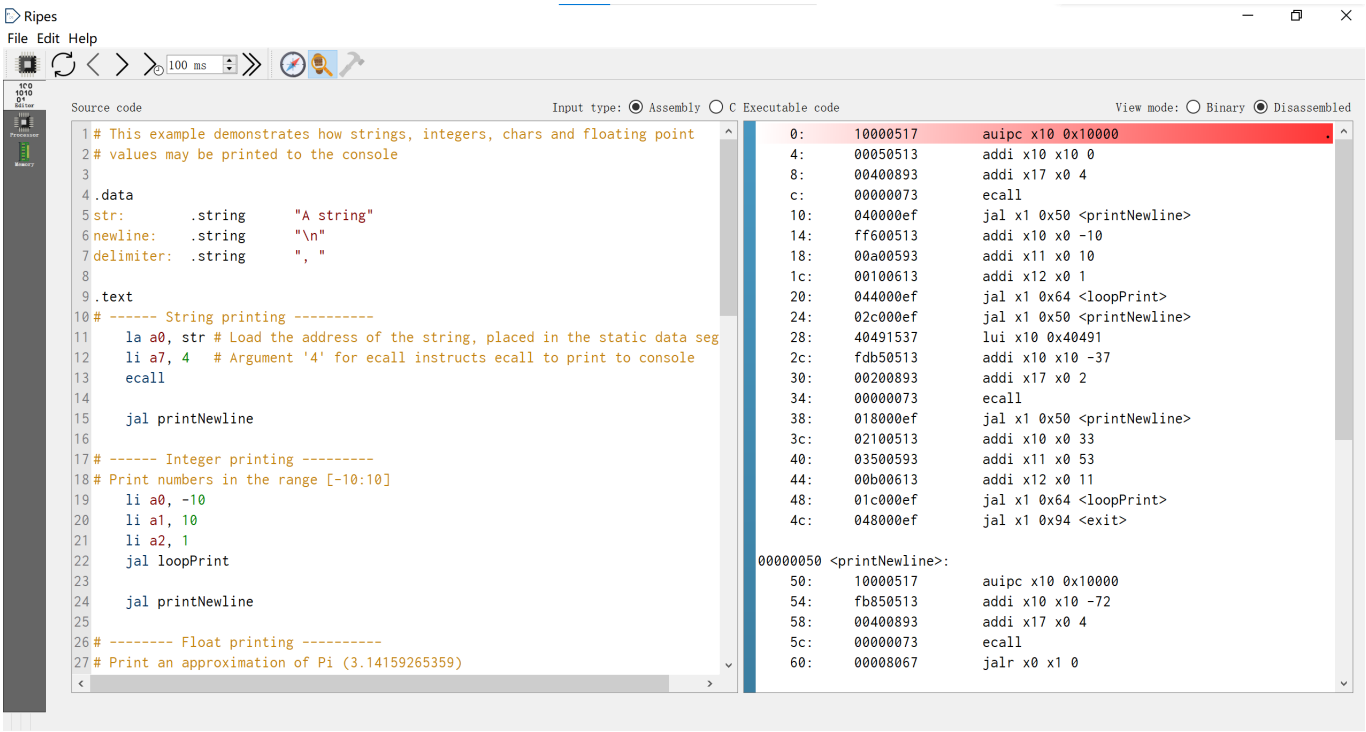
三、实验平台

- PC 一台
- Ripes： RISC-V graphical processor simulator
- Rars：RISC-V Assembler and Runtime Simulator

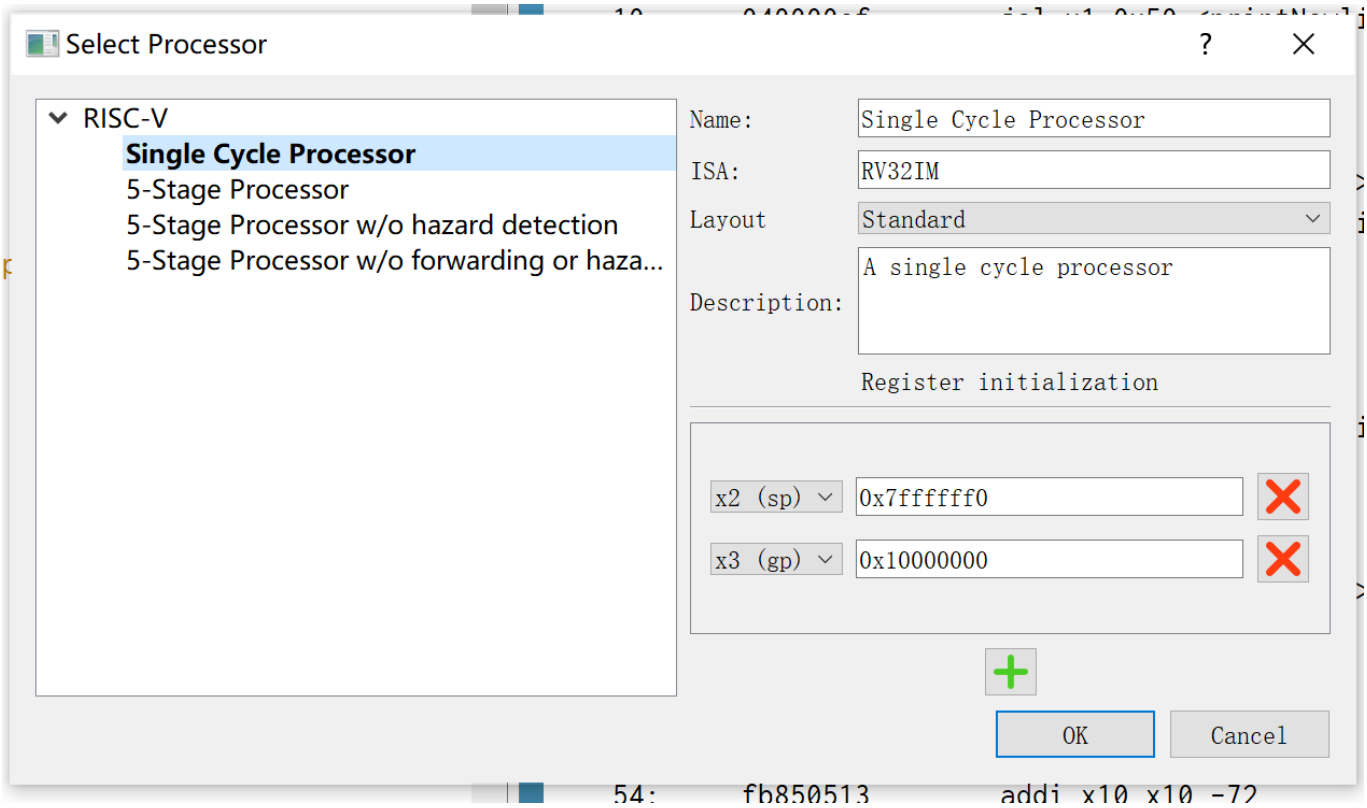
四、实验过程

理解并仿真RIPES示例汇编程序

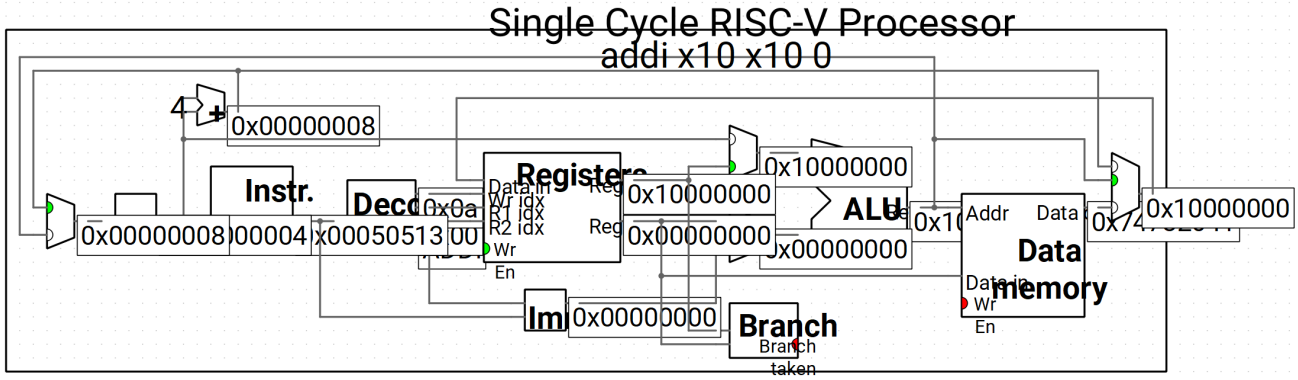
首先在ripes中打开console printing函数：



之后选择单周期cpu数据通路：



单步执行指令观察数据通路，下图为addi指令：



Rars软件设计汇编程序，实现人工检查6条指令功能，并生成COE文件

根据ppt中的示例代码可以得到以下指令功能验证：

- sw指令：

```
sw x0, 0(a0)                #test sw: 全灭led
```

- lw指令：

```
lw t0, 4(a0)                #test lw: 由switch设置led
sw t0, 0(a0)
```

- addi指令：

```
addi t0, x0, 0xff       #test addi: 全亮led
sw t0, 0(a0)
```

- add指令：

```
andi t0, x0, 0x00       #test add: 半亮前led
addi t1, x0, 0x0f
add t0, x0, t1
sw t0, 0(a0)
```

- beq指令：

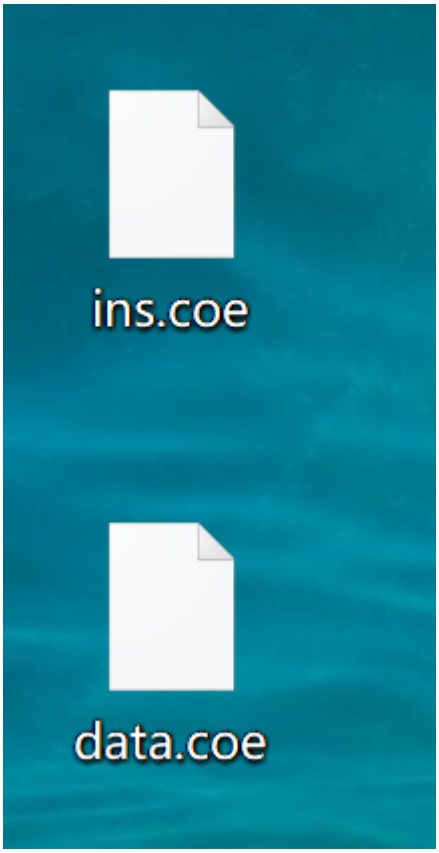
```
beq t0, t1, sign1       #test beq: 半亮后led
sign1:
addi t0, x0, 0xf0
sw t0, 0(a0)
```

beq t1,t2,label Bran

- jal 指令：

```
jal x1, sign2           #test jal: 半亮中led
sign2:
addi t0, x0, 0x3c
sw t0, 0(a0)
```

生成的coe文件如下：

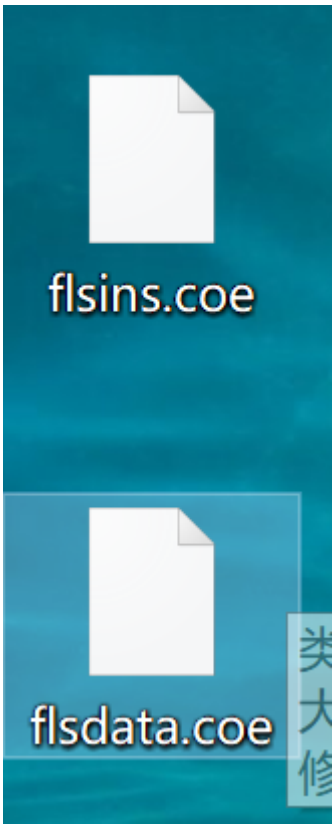


Rars软件设计汇编程序，实现计算斐波那契—卢卡斯数列（数列前两项为1，2），并生成COE文件

设计出的汇编文件如下：

```
        lw a0, number
        addi a1, x0, 1
        addi a2, x0, 2
        beq a0, a1, edge1
        beq a0, a2, edge2
        addi a4, x0, 3
loop:
        add a3, a2, a1
        beq a4, a0, edge3
        addi a4, a4, 1
        add a1, a2, x0
        add a2, a3, x0
        beq x0, x0, loop
edge1:
        add a3, x0, a1
        beq x0, x0, edge3
edge2:
        add a3, x0, a2
edge3:
```

生成的coe文件如下：



五、实验结果

理解并仿真RIPES示例汇编程序

标准输出程序运行结果如下：

```
Console

A string

-10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
3.14159
!, ", #, $, %, &, ', (, ), *, +, ,, -, ., /, 0, 1, 2, 3, 4, 5,
Program exited with code: 0
```

Rars软件设计汇编程序，实现人工检查6条指令功能，并生成COE文件

初始时led全亮，如下：

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00003000	0xffffd517	auipc x10,0xffffffffd	6: la a0, out #仿真需要
	0x00003004	0x00050513	addi x10,x10,0	
	0x00003008	0x00052023	sw x0,0(x10)	8: sw x0, 0(a0) #test sw: 全灭led
	0x0000300c	0x0ff00293	addi x5,x0,0x000000ff	10: addi t0, x0, 0xff #test addi: 全亮led
	0x00003010	0x00552023	sw x5,0(x10)	11: sw t0, 0(a0)
	0x00003014	0x00452283	lw x5,4(x10)	13: lw t0, 4(a0) #test lw: 由switch设置led
	0x00003018	0x00552023	sw x5,0(x10)	14: sw t0, 0(a0)
	0x0000301c	0x00007293	andi x5,x0,0	16: andi t0, x0, 0x00 #test add: 半亮前led
	0x00003020	0x00f00313	addi x6,x0,15	17: addi t1, x0, 0x0f

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00000000	0x000000ff	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

sw执行完，可以发现对应地址清零，成功将led全灭：

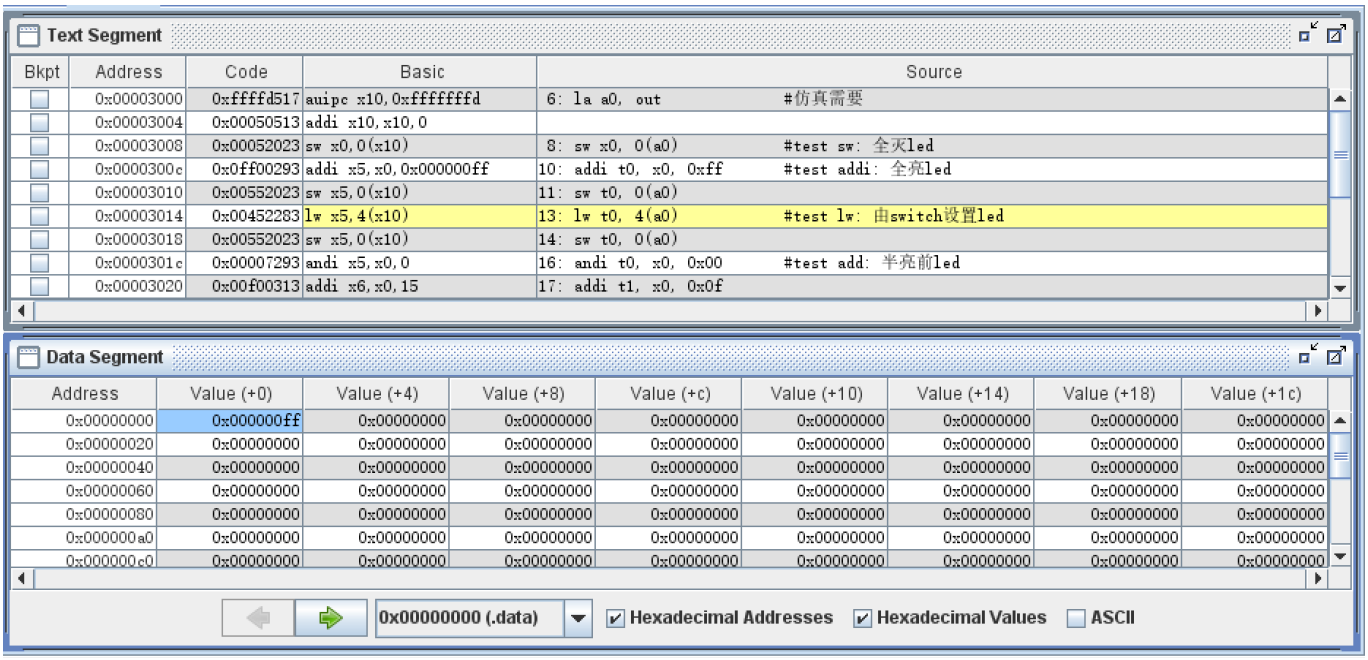
Text Segment

Bkpt	Address	Code	Basic	Source
	0x00003000	0xffffd517	auipc x10,0xffffffffd	6: la a0, out #仿真需要
	0x00003004	0x00050513	addi x10,x10,0	
	0x00003008	0x00052023	sw x0,0(x10)	8: sw x0, 0(a0) #test sw: 全灭led
	0x0000300c	0x0ff00293	addi x5,x0,0x000000ff	10: addi t0, x0, 0xff #test addi: 全亮led
	0x00003010	0x00552023	sw x5,0(x10)	11: sw t0, 0(a0)
	0x00003014	0x00452283	lw x5,4(x10)	13: lw t0, 4(a0) #test lw: 由switch设置led
	0x00003018	0x00552023	sw x5,0(x10)	14: sw t0, 0(a0)
	0x0000301c	0x00007293	andi x5,x0,0	16: andi t0, x0, 0x00 #test add: 半亮前led
	0x00003020	0x00f00313	addi x6,x0,15	17: addi t1, x0, 0x0f

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

addi执行完， 可以发现对应地址赋值0xff，成功将所有led点亮：



Text Segment

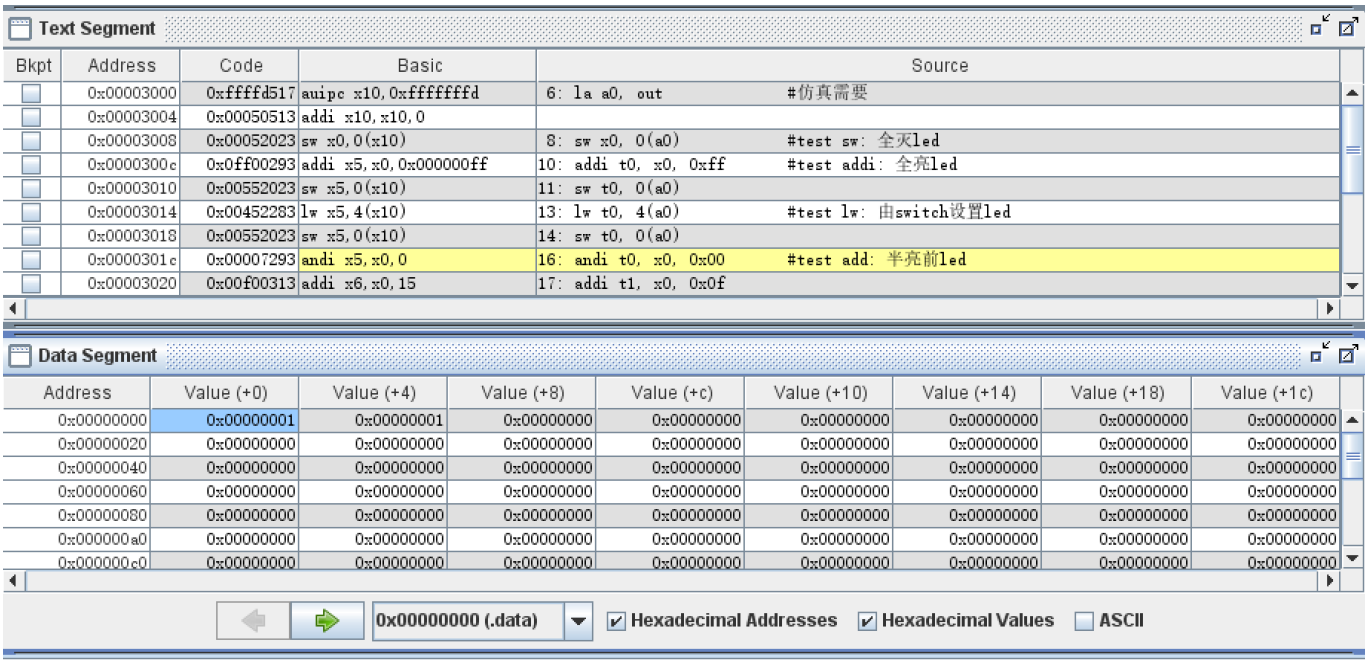
Bkpt	Address	Code	Basic	Source
	0x00003000	0xffffd517	auipc x10,0xffffffff	6: la a0, out #仿真需要
	0x00003004	0x00050513	addi x10,x10,0	
	0x00003008	0x00052023	sw x0,0(x10)	8: sw x0, 0(a0) #test sw: 全灭led
	0x0000300c	0x0ff00293	addi x5,x0,0x000000ff	10: addi t0, x0, 0xff #test addi: 全亮led
	0x00003010	0x00552023	sw x5,0(x10)	11: sw t0, 0(a0)
	0x00003014	0x00452283	lw x5,4(x10)	13: lw t0, 4(a0) #test lw: 由switch设置led
	0x00003018	0x00552023	sw x5,0(x10)	14: sw t0, 0(a0)
	0x0000301c	0x00007293	andi x5,x0,0	16: andi t0, x0, 0x00 #test add: 半亮前led
	0x00003020	0x00f00313	addi x6,x0,15	17: addi t1, x0, 0x0f

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00000000	0x000000ff	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

0x00000000 (.data) Hexadecimal Addresses Hexadecimal Values ASCII

lw执行完，可以看出led按照in中sw输入的值进行赋值：



Text Segment

Bkpt	Address	Code	Basic	Source
	0x00003000	0xffffd517	auipc x10,0xffffffff	6: la a0, out #仿真需要
	0x00003004	0x00050513	addi x10,x10,0	
	0x00003008	0x00052023	sw x0,0(x10)	8: sw x0, 0(a0) #test sw: 全灭led
	0x0000300c	0x0ff00293	addi x5,x0,0x000000ff	10: addi t0, x0, 0xff #test addi: 全亮led
	0x00003010	0x00552023	sw x5,0(x10)	11: sw t0, 0(a0)
	0x00003014	0x00452283	lw x5,4(x10)	13: lw t0, 4(a0) #test lw: 由switch设置led
	0x00003018	0x00552023	sw x5,0(x10)	14: sw t0, 0(a0)
	0x0000301c	0x00007293	andi x5,x0,0	16: andi t0, x0, 0x00 #test add: 半亮前led
	0x00003020	0x00f00313	addi x6,x0,15	17: addi t1, x0, 0x0f

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00000000	0x00000001	0x00000001	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

0x00000000 (.data) Hexadecimal Addresses Hexadecimal Values ASCII

add执行完，可以看出前四个led亮了起来：

Text Segment

Bkpt	Address	Code	Basic	Source
	0x0000300c	0x0ff00293	addi x5, x0, 0x000000ff	10: addi t0, x0, 0xff #test addi: 全亮led
	0x00003010	0x00552023	sw x5, 0(x10)	11: sw t0, 0(a0)
	0x00003014	0x00452283	lw x5, 4(x10)	13: lw t0, 4(a0) #test lw: 由switch设置led
	0x00003018	0x00552023	sw x5, 0(x10)	14: sw t0, 0(a0)
	0x0000301c	0x00007293	andi x5, x0, 0	16: andi t0, x0, 0x00 #test add: 半亮前led
	0x00003020	0x00f00313	addi x6, x0, 15	17: addi t1, x0, 0x0f
	0x00003024	0x006002b3	add x5, x0, x6	18: add t0, x0, t1
	0x00003028	0x00552023	sw x5, 0(x10)	19: sw t0, 0(a0)
	0x0000302c	0x00628263	beq x5, x6, 0x00000004	21: beq t0, t1, sign1 #test beq: 半亮后led

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00000000	0x0000000f	0x00000001	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

beq会跳转至sign1位置，并让后四个led亮起来：

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00003018	0x00552023	sw x5, 0(x10)	14: sw t0, 0(a0)
	0x0000301c	0x00007293	andi x5, x0, 0	16: andi t0, x0, 0x00 #test add: 半亮前led
	0x00003020	0x00f00313	addi x6, x0, 15	17: addi t1, x0, 0x0f
	0x00003024	0x006002b3	add x5, x0, x6	18: add t0, x0, t1
	0x00003028	0x00552023	sw x5, 0(x10)	19: sw t0, 0(a0)
	0x0000302c	0x00628263	beq x5, x6, 0x00000004	21: beq t0, t1, sign1 #test beq: 半亮后led
	0x00003030	0x0f000293	addi x5, x0, 0x000000f0	23: addi t0, x0, 0xf0
	0x00003034	0x00552023	sw x5, 0(x10)	24: sw t0, 0(a0)
	0x00003038	0x004000ef	jal x1, 0x00000004	26: jal x1, sign2 #test jal: 半亮中led

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00000000	0x000000f0	0x00000001	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

jal会跳转到sign2位置，并让中间四个led亮起来：

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00003020	0x00f00313	addi x6, x0, 15	17: addi t1, x0, 0x0f
	0x00003024	0x006002b3	add x5, x0, x6	18: add t0, x0, t1
	0x00003028	0x00552023	sw x5, 0(x10)	19: sw t0, 0(a0)
	0x0000302c	0x00628263	beq x5, x6, 0x00000004	21: beq t0, t1, sign1 #test beq: 半亮后led
	0x00003030	0x0f000293	addi x5, x0, 0x000000f0	23: addi t0, x0, 0xf0
	0x00003034	0x00552023	sw x5, 0(x10)	24: sw t0, 0(a0)
	0x00003038	0x004000ef	jal x1, 0x00000004	26: jal x1, sign2 #test jal: 半亮中led
	0x0000303c	0x03c00293	addi x5, x0, 0x0000003c	28: addi t0, x0, 0x3c
	0x00003040	0x00552023	sw x5, 0(x10)	29: sw t0, 0(a0)

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00000000	0x0000003c	0x00000001	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Rars软件设计汇编程序，实现计算斐波那契—卢卡斯数列（数列前两项为1，2），并生成COE文件

该程序提取了.data端中所需数字n从而在a3数组中存储得到第f_n项的值

Data Segment

Address	Value (+0)
0x00000000	0x00000005

运行结果如下输出$f_5 = 8$

a0	10	0x00000005
a1	11	0x00000003
a2	12	0x00000005
a3	13	0x00000008
a4	14	0x00000005
a5	15	0x00000000

六、心得体会

本实验进一步让我熟悉了汇编语言的使用，并且做好了今后cpu调试的准备工作。本试验任务量一般，在课内实验使用ripes的基础上，我学会了rars软件的使用和生成coe文件，收获还是挺大的。