

# Lab 1 report

PB22051022 王嘉宁

## 实验目的与内容

熟悉 RISC 指令集（龙芯 LA32R 指令集以及 RV32I 指令集）的部分指令及编码格式，以便后续在硬件层面上实现这些指令。

## 逻辑设计

任务 1：斐波那契数列

```
li.w $r2, 9
andi $r1, $r1, 0
andi $r3, $r3, 0
andi $r4, $r4, 0
andi $r5, $r5, 0
addi.w $r3, $r3, 1
addi.w $r4, $r4, 1
addi.w $r1, $r1, 2
a:
bge $r1, $r2, exit
addi.w $r5, $r4, 0
addi.w $r4, $r3, 0
add.w $r3, $r4, $r5
addi.w $r1, $r1, 1
b a
exit:
addi.w $r3, $r3, 0
```

任务 2：大整数处理

```
li.w $r2, 80
andi $r1, $r1, 0
andi $r3, $r3, 0
andi $r4, $r4, 0
andi $r5, $r5, 0
andi $r6, $r6, 0
andi $r7, $r7, 0
andi $r8, $r8, 0
addi.w $r4, $r4, 1
addi.w $r6, $r6, 1
addi.w $r1, $r1, 2
a:
bge $r1, $r2, exit
```

```

andi $r9, $r9, 0
addi.w $r8, $r6, 0
addi.w $r6, $r4, 0
add.w $r4, $r6, $r8
bgeu $r4, $r6, m
addi.w $r9, $r9, 1
m:
addi.w $r7, $r5, 0
addi.w $r5, $r3, 0
add.w $r3, $r5, $r7
add.w $r3, $r3, $r9
addi.w $r1, $r1, 1
b a
exit:
addi.w $r3, $r3, 0
addi.w $r4, $r4, 0

```

## 结果与分析

### 任务 1：斐波那契数列

The screenshot shows the LARS debugger interface. On the left, the assembly code for the Fibonacci sequence calculation is displayed:

```

li.w $r2, 9
andi $r1, $r1, 0
andi $r3, $r3, 0
andi $r4, $r4, 0
andi $r5, $r5, 0
addi.w $r3, $r3, 1
addi.w $r4, $r4, 1
addi.w $r1, $r1, 1
a:
bge $r1, $r2, exit
addi.w $r5, $r4, 0
addi.w $r4, $r3, 0
add.w $r3, $r4, $r5
addi.w $r1, $r1, 1
b a
exit:
addi.w $r3, $r3, 0

```

In the center, the register state is shown in a grid:

PC 0x1c00003c		Inst NOP	
R0 / ZERO 0x0	R1 / RA 0x9	R2 / TP 0x9	R3 / SP 0x22
R4 / A0 0x15	R5 / A1 0xd	R6 / A2 0x0	R7 / A3 0x0
R8 / A4 0x0	R9 / A5 0x0	R10 / A6 0x0	R11 / A7 0x0
R12 / T0 0x0	R13 / T1 0x0	R14 / T2 0x0	R15 / T3 0x0
R16 / T4 0x0	R17 / T5 0x0	R18 / T6 0x0	R19 / T7 0x0

On the right, a memory dump table is shown:

Memory	+0	+4	+8	+c
0x1c000000	0x03802402	0x03400021	0x03400063	0x03400084
0x1c000010	0x03400045	0x02800463	0x02800484	0x02800821
0x1c000020	0x64001822	0x02800085	0x02800064	0x00101483
0x1c000030	0x02800421	0x53feffff	0x02800063	0x00000000
0x1c000040	0x00000000	0x00000000	0x00000000	0x00000000
0x1c000050	0x00000000	0x00000000	0x00000000	0x00000000
0x1c000060	0x00000000	0x00000000	0x00000000	0x00000000
0x1c000070	0x00000000	0x00000000	0x00000000	0x00000000
0x1c000080	0x00000000	0x00000000	0x00000000	0x00000000
0x1c000090	0x00000000	0x00000000	0x00000000	0x00000000
0x1c0000a0	0x00000000	0x00000000	0x00000000	0x00000000

## 任务 2：大整数处理

The screenshot shows the LARS assembly editor interface. On the left, the assembly code is displayed:

```

li.w $r2, 80
andi $r1, $r1, 0
andi $r3, $r3, 0
andi $r4, $r4, 0
andi $r5, $r5, 0
andi $r6, $r6, 0
andi $r7, $r7, 0
andi $r8, $r8, 0
addi.w $r4, $r4, 1
addi.w $r6, $r6, 1
addi.w $r1, $r1, 2
a:
bge $r1, $r2, exit
andi $r9, $r9, 0
addi.w $r8, $r6, 0
addi.w $r6, $r4, 0
add.w $r4, $r6, $r8
bgeu $r4, $r6, m
addi.w $r9, $r9, 1
n:
addi.w $r7, $r5, 0
addi.w $r5, $r3, 0
add.w $r3, $r5, $r7
add.w $r3, $r3, $r9
addi.w $r1, $r1, 1
b a:
exit:
addi.w $r3, $r3, 0
addi.w $r4, $r4, 0

```

On the right, the register state is shown in a grid:

PC 0x1c000068		Inst NOP	
R0 / ZERO 0x0	R1 / RA 0x50	R2 / TP 0x50	R3 / SP 0x533163
R4 / A0 0x0ef0321e5	R5 / A1 0x336a82	R6 / A2 0xb89c937d	R7 / A3 0x1fc6e1
R8 / A4 0x16668e68	R9 / A5 0x0	R10 / A6 0x0	R11 / A7 0x0
R12 / T0 0x0	R13 / T1 0x0	R14 / T2 0x0	R15 / T3 0x0
R16 / T4 0x0	R17 / T5 0x0	R18 / T6 0x0	R19 / T7 0x0

Below the registers is a memory dump table:

Memory	+0	+4	+8	+c
0x1c000000	0x03814002	0x03400021	0x03400063	0x03400084
0x1c000010	0x034000a5	0x034000c6	0x034000e7	0x03400108
0x1c000020	0x02800484	0x028004c6	0x02800821	0x64003422
0x1c000030	0x03400129	0x028000c8	0x02800086	0x001020c4
0x1c000040	0x6c000886	0x02800529	0x02800a07	0x02800065
0x1c000050	0x00101ca3	0x00102463	0x02800421	0x53ffef3ff
0x1c000060	0x02800063	0x02800084	0x00000000	0x00000000
0x1c000070	0x00000000	0x00000000	0x00000000	0x00000000
0x1c000080	0x00000000	0x00000000	0x00000000	0x00000000
0x1c000090	0x00000000	0x00000000	0x00000000	0x00000000
0x1c0000a0	0x00000000	0x00000000	0x00000000	0x00000000

## 任务 3：导出 COE 文件

### 任务一

```
C: > Users > Lenovo > Desktop > t1.coe
1   memory_initialization_radix=16;
2   memory_initialization_vector=
3   03802402,
4   03400021,
5   03400063,
6   03400084,
7   034000a5,
8   02800463,
9   02800484,
10  02800821,
11  64001822,
12  02800085,
13  02800064,
14  00101483,
15  02800421,
16  53ffffff,
17  02800063,
18
```

## 任务二

The screenshot shows a terminal window with the following interface elements:

- Top left: A blue vertical bar.
- Top center: A tab labeled "预览 li.md".
- Top right: A tab labeled "t1.coe" and another tab labeled "t2.coe" which is currently active.
- Close button: A standard "X" icon.

The main content area displays the following text:

```
C: > Users > Lenovo > Desktop > t2.coe
1   memory_initialization_radix=16;
2   memory_initialization_vector=
3   03814002,
4   03400021,
5   03400063,
6   03400084,
7   034000a5,
8   034000c6,
9   034000e7,
10  03400108,
11  02800484,
12  028004c6,
13  02800821,
14  64003422,
15  03400129,
16  028000c8,
17  02800086,
18  001020c4,
19  6c000886,
20  02800529,
21  028000a7,
22  02800065,
23  00101ca3,
24  00102463,
25  02800421,
26  53ffd3ff,
27  02800063,
28  02800084,
29
```

## 总结

对龙芯 LA32R 指令集部分指令及编码格式有所熟悉，并掌握了一些使用软件基本技巧。