

lab4 实验报告

李远航

PB20000137

1. 实验内容

- task1

阅读并补全代码，使得运行之后寄存器状态为：

```
1 R0 = 5, R1 = 0, R2 = 300f, R3 = 0
2 R4 = 0, R5 = 0, R6 = 0, R7 = 3003
```

- task2

阅读代码，补全求一个数mod 7的余数的代码

2. 完成过程

- task1

```
1 1110010000001110
2 0101000000100000
3 0100100000000001
4 1111000000100101
5 0111111010000000
6 0001010010100001
7 0001000000100001
8 0010001000010001
9 0001001001111111
10 0011001000001111
11 0000010000000001
12 0100111111111000
13 0001010010111111
14 0110111010000000
15 1100000111000000
16 0000000000000000
17 0000000000000000
18 0000000000000000
19 0000000000000000
20 0000000000000000
21 0000000000000000
22 0000000000000000
23 0000000000000000
24 0000000000000000
25 0000000000000000
26 0000000000000101
```

- 第一处的缺失显然是 1，否则，程序将直接结束
- 第二处缺失根据最后程序最后一行储存的数，以及程序最后的寄存器状态，可以得到为 0
- 第三处缺失显然应该对 R1 进行操作，因此应该填 0
- 第四处缺失是要给 R7 赋值，以便于后续的跳转，所以应该填 1

运行结果如下所示，符合题目要求

```
R0 = 5, R1 = 0, R2 = 300f, R3 = 0
R4 = 0, R5 = 0, R6 = 0, R7 = 3003
```

- **task2**

task2是求mod 7 余数的程序，主要思路:

$$X \equiv 8 \times x + R \equiv x + R \pmod{7}$$

因此只要将代求数字 X 除以8的商和余数相加得到一个新的数字，不断重复同样的操作，直到这个数字小于7，便可以得到答案

```
1 0010001000010101
2 0100100000001000
3 0101010001100111
4 0001001010000100
5 0001000001111001
6 0000001111111011
7 0001000001111001
8 0000100000000001
9 0001001001111001
10 1111000000100101
11 0101010010100000
12 0101011011100000
13 0101100100100000
14 0001010010100001
15 0001011011101000
16 0101101011000001
17 0000010000000001
18 0001100010000100
19 0001010010000010
20 0001011011000011
21 0000001111111010
22 1100000111000000
23 0000000100100000
```

- 第一处根据上述分析，为判断求出的数字是否满足小于7的部分，同时这是对立即数进行操作，所以填入011
- 第二处为跳转部分，为大于7的情况下重新执行函数，所以填入111
- 第三处目的是最后计算出来的数是7，则通过判断，将最后的答案变成0，填入011
- 第四处为计算除以8的商的部分，目的是左移一位，所以填入011
- 第五处为判断是否计算完成，当上一步移位仍然大于0时，应该继续循环填入001

运行结果如下所示，R1 寄存器储存 $288 \equiv 1 \pmod{7}$:

```
R0 = fffa, R1 = 1, R2 = 0, R3 = 8000
R4 = 1, R5 = 0, R6 = 0, R7 = 3002
```

3. 实验收获

- 更加熟练了阅读 1c-3 汇编码的能力
- 学会利用相关数学知识，在 1c-3 指令集下，获得一些特殊数字余数的方法