

Lab3 Better Angels

PB20000248 郑子涵

实验要求

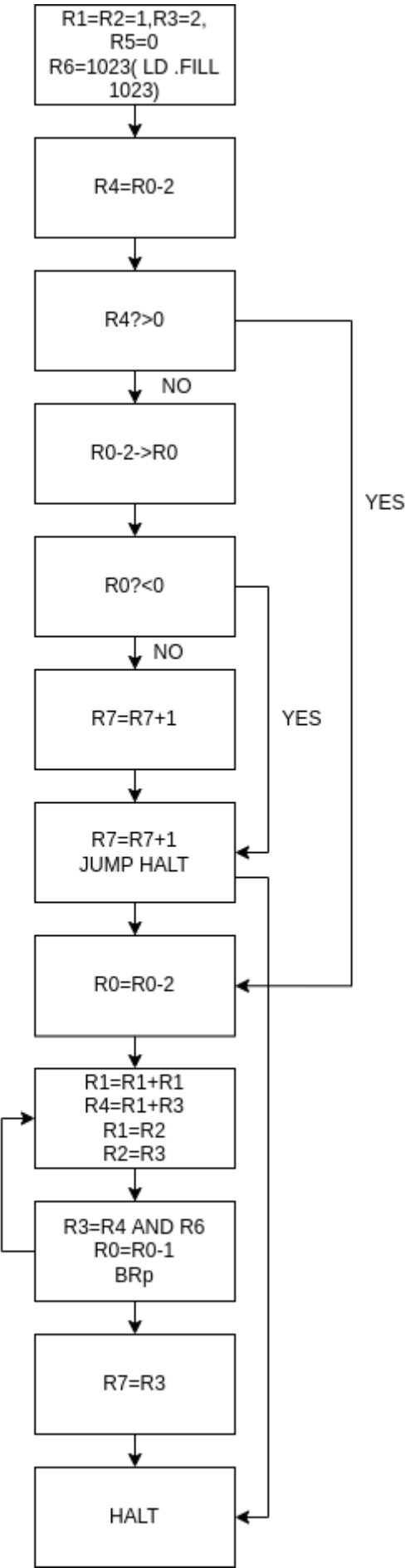
- **read and understand**: 读懂代码
- **guess**: 根据代码的后四行来猜测该代码拥有者的学号
- **optimize**: 分配的代码是一个L版本的代码，需要优化其代码，使得执行的指令数减少，即完成一个P版本的代码
- **(optional): feedback**: 可以向代码的拥有者反馈

实验过程

1: Read and understand

初始代码如下:

```
.ORIG x3000
ADD R1, R1, #1 ;初始化R1, 存放F(0)
ADD R2, R2, #1 ;初始化R2, 存放F(1)
ADD R3, R3, #2 ;初始化R3, 存放F(2)
LD R6, Num ;存放1023于R6, 用于mod操作
ADD R4, R0, #-2 ;判断R0是否大于2
BRp Lable0 ;大于2, 跳转
ADD R0, R0, #-2 ;否则, 继续判断是否小于2
BRn #1 ;小于2 (为0或1), 少加一次1
ADD R7, R7, #1
ADD R7, R7, #1
BRnzp EXIT
Lable0 ADD R0, R0, #-2 ;R0减2, R0存放循环次数
Lable1 ADD R1, R1, R1 ;F(n-3)*2
ADD R4, R1, R3 ;F(n-3)*2+F(n-1)
ADD R1, R2, #0 ;R1存放F(n-2)
ADD R2, R3, #0 ;R2存放F(n-1)
AND R3, R4, R6 ;(F(n-3)*2+F(n-1))mod1024=F(n)
ADD R0, R0, #-1 ;循环次数减1
BRp Lable1 ;判断循环是否结束
ADD R7, R3, #0 ;存放F(n)于R3
EXIT HALT
Num .FILL #1023
Fa .FILL #930 ;F(20)
Fb .FILL #30 ;F(07)
Fc .FILL #178 ;F(14)
Fd .FILL #710 ;F(13)
.END
```



根据上面的代码画出流程图如下（理解代码的结构）：

2: Guess

根据程序最后四行的.FILL伪代码得到 $F(a)=930, F(b)=30, F(c)=178, F(d)=710$ 。先根据学号的不同位代表的含义，和数列值的大小，缩小一下猜测的范围，再通过求数列的程序一个一个尝试，便可求得学号，最终求得结果如下： $a=20, b=7, c=14, d=13$ 。即学号为**PB20071413**。另一种方法，可以稍微改写一下lab2中的程序，使得其可以返回第一个数列中出现该数字的LC3程序。实现如下（因为为0，1，2时，不用程序也容易直接猜出，所以下程序省略了，那三种情况）：

```
.ORIG    x3000
    ADD R1, R1, #1
    ADD R2, R2, #1
    ADD R3, R3, #2
    ADD R5, R5, #2
    NOT R0, R0
    ADD R0, R0, #1
    LD  R4, NUM
POSE3 ADD    R1, R1, R1
    ADD R7, R1, R3
    AND R7, R7, R4
    ADD R1, R2, #0
    ADD R2, R3, #0
    ADD R3, R7, #0
    ADD R5, R5, #1
    ADD R7, R7, R0
    BRnp POSE3
    HALT
    NUM .FILL #1023
.END
```

3: optimize

通过对数列的观察，可以发现数列是一个循环的数列，周期为128，从 $F(20)$ 开始每过128个数之后边开始循环，所以我们只需判断 $F(0)-F(19)$ ，并把之后的数都归并到 $F(20)-F(147)$ 再判断即可，这样便可以减少指令执行的条数。其实现就是在lab2的基础上多加一步，来把R0的范围归纳到0—147。具体代码如下：

```
.ORIG x3000
ADD R0, R0, #-10
ADD R0, R0, #-10
BRn POSE2
POSE4 LD R6 NUM5
ADD R0, R0, R6
BRzp POSE4
LD R7, NUM3
ADD R0, R0, R7
POSE2 ADD    R0, R0, #10
ADD R0, R0, #10
ADD R1, R1, #1
ADD R2, R2, #1
ADD R3, R3, #2
LD R6, Num
ADD R4, R0, #-2
```

```
BRp Lable0
ADD R0, R0, #-2
BRn #1
ADD R7, R7, #1
ADD R7, R7, #1
BRnzp EXIT
Lable0 ADD R0, R0, #-2
Lable1 ADD R1, R1, R1
ADD R4, R1, R3
ADD R1, R2, #0
ADD R2, R3, #0
AND R3, R4, R6
ADD R0, R0, #-1
BRp Lable1
ADD R7, R3, #0
EXIT HALT
Num .FILL #1023
NUM5 .FILL #-128
NUM3 .FILL #128
.END
```

实验总结

- 对于我自己的lab2中的代码在样例测试中的平均指令数为：43302
- 对于该代码的原始版本在样例测试中的平均指令数为：30298
- 对于优化过后的代码在样例测试中的平均指令数为：660
- 优化过后使得指令数变成了约为原来的**2.2%**，更是我原本指令数的**1.5%**,大大减少了指令执行的条数，同时也并没有过于增加指令的条数。