

# lab2:Fibonacci sequence

---

郑子涵 PB20000248

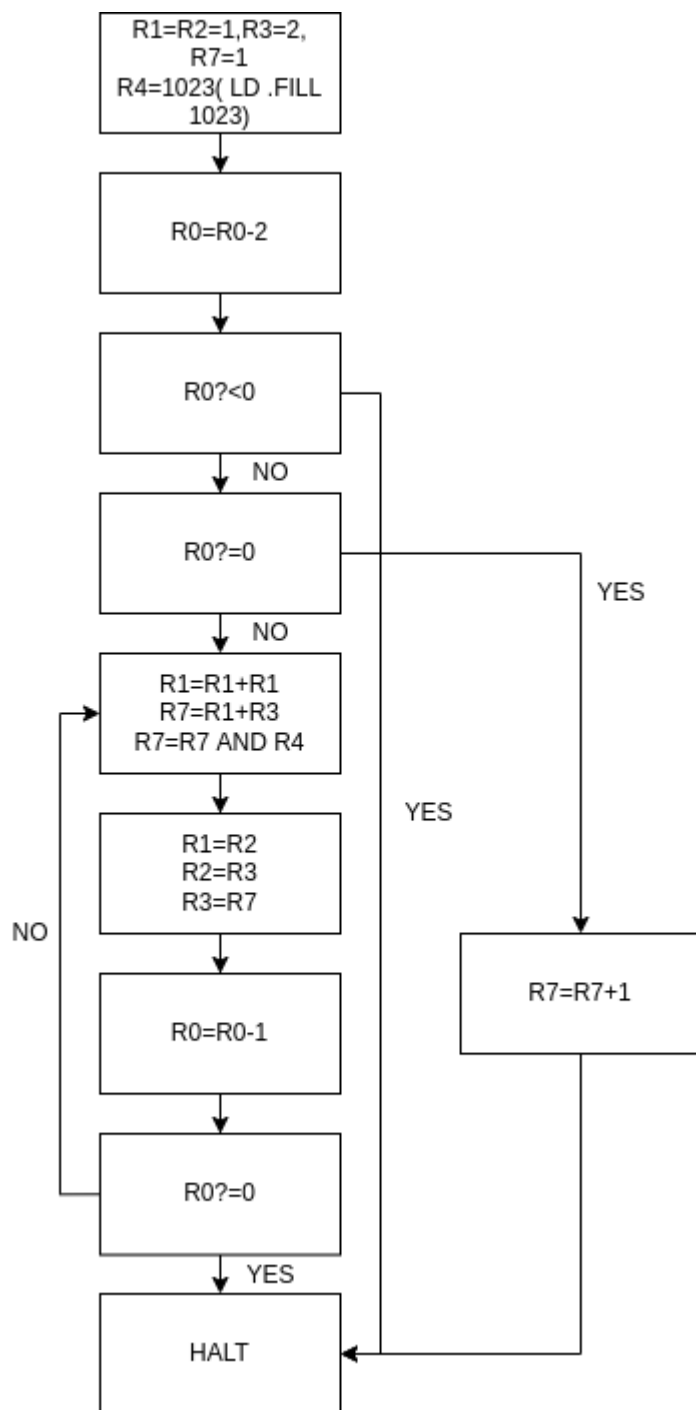
## 实验要求

- 把一个数列的第 $n$ 位存储到**R7**寄存器.
- 数列满足如下条件:  $F(0) = 1$   $F(1) = 1$   $F(2) = 2$  .....  $F(n) = (F(n-1) + 2 * F(n-3)) \bmod 1024$  ( $1 \leq n \leq 16384$ )
- 最开始 **$n$ 储存在R0，其余寄存器均为0.**
- 把学号每两位分开为**a, b, c, d**四部分，如PB20000248，有 $a=20$ ， $b=0$ ， $c=2$ ， $d=48$ 。用".FILL"伪代码储存  $F(a)$ ,  $F(b)$ ,  $F(c)$ ,  $F(d)$  的值在代码的结尾。

## 设计实现

### 设计思路

按照数列的定义来计算 $F(n)$ 的值，把 $F(0)$ ， $F(1)$ ， $F(2)$  分开来讨论，其余的按定义依次计算出每个值，取余操作通过"AND 1023"来实现，即保留小于1024的部分，一直到 $n$ 为止。再通过运行程序，算出 $F(a)$ ,  $F(b)$ ,  $F(c)$ ,  $F(d)$  的值后，用".FILL"伪代码填入程序中 流程图如下：



初始代码

核心代码**23**行

```

.ORIG    x3000
    AND R1, R1, #0
    AND R2, R2, #0
    AND R3, R3, #0
    AND R4, R4, #0
    AND R7, R7, #0
    ADD R1, R1, #1
    ADD R2, R2, #1
    ADD R3, R3, #2
    ADD R7, R7, #1
  
```

```

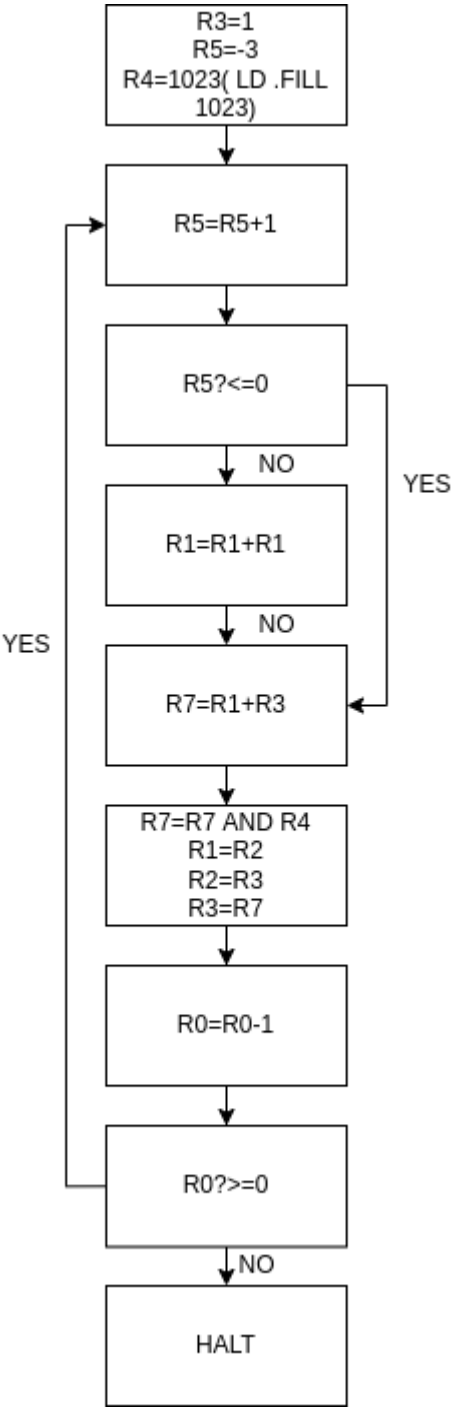
LD  R4, NUM
ADD R0, R0, #-2
BRn END1
BRz POSE1
POSE3 ADD  R1, R1, R1
ADD R7, R1, R3
AND R7, R7, R4
ADD R1, R2, #0
ADD R2, R3, #0
ADD R3, R7, #0
ADD R0, R0, #-1
BRp POSE3
BRnzp  END1
POSE1 ADD  R7, R7, #1
END1 HALT
NUM .FILL #1023
a .FILL #930;F(a),a=20
b .FILL #1;F(b),b=0
c .FILL #2;f(c),c=2
d .FILL #898;F(d),d=48
.END

```

## 优化思路

刚开始想通过设置 $F(-3)$ , $F(-2)$ , $F(-1)$ 来把 $F(0)$ ,  $F(1)$ ,  $F(2)$ 的情况包括到一般的情况中，但发现会出现浮点数（ $F(-1)=1/2$ , $F(-2)=0$ , $F(-3)=1/4$ ），用LC3难以实现。但可以通过改变前三次的递推规则来使结果成立。具体如

下：  $F(-3)=0$   $F(-2)=0$   $F(-1)=1$   $F(n) = (F(n-1) + F(n-3)) \bmod 1024$  ( $0 \leq n \leq 2$ ) 如此便可使这几种情况有所合并 流



程图如下：

优化代码

核心代码**18**行

```
.ORIG    x3000
    AND R1, R1, #0
    AND R2, R2, #0
    AND R3, R3, #0
    AND R4, R4, #0
    AND R7, R7, #0
    ADD R3, R3, #1
    AND R5, R5, #0
    ADD R5, R5, #-3
```

```

LD  R4, NUM
POSE3 ADD  R5, R5, #1
BRnz POSE1
ADD R1, R1, R1
POSE1 ADD  R7, R1, R3
AND R7, R7, R4
ADD R1, R2, #0
ADD R2, R3, #0
ADD R3, R7, #0
ADD R0, R0, #-1
BRzp POSE3
END1 HALT
NUM .FILL #1023
a .FILL #930;F(a),a=20
b .FILL #1;F(b),b=0
c .FILL #2;f(c),c=2
d .FILL #898;F(d),d=48
.END

```

## 实验总结

最开始是按分类讨论的思想来写，但为了优化代码行数，试图把这几种情况合并在一起，但是需要稍微改变规则才能完成，但总归还是用递推的思想来写的。

## 样例测试

- 把样例运行的结果和编写的C语言运行结果相比较即可，若相同则成功。
- F(48)=898

Registers			Memory		
R0	xFFFF	65535	① ▶ x3000	x5260	21088 AND R1, R1, #0
R1	x03B2	946	① ▶ x3001	x54A0	21664 AND R2, R2, #0
R2	x0076	118	① ▶ x3002	x56E0	22240 AND R3, R3, #0
R3	x0382	898	① ▶ x3003	x5920	22816 AND R4, R4, #0
R4	x03FF	1023	① ▶ x3004	x5FE0	24544 AND R7, R7, #0
R5	x002E	46	① ▶ x3005	x16E1	5857 ADD R3, R3, #1
R6	x0000	0	① ▶ x3006	x5B60	23392 AND R5, R5, #0
R7	x0382	898	① ▶ x3007	x1B7D	7037 ADD R5, R5, #-3
PSR	x8004	32772 CC: N	① ▶ x3008	x280B	10251 LD R4, NUM
PC	x3013	12307	① ▶ x3009	x1B61	7009 POSE3 ADD R5, R5, #1
MCR	x0000	0	① ▶ x300A	x0C01	3073 BRnz POSE1
Console (click to focus)			① ▶ x300B	x1241	4673 ADD R1, R1, R1
			① ▶ x300C	x1E43	7747 POSE1 ADD R7, R1, R3
			① ▶ x300D	x5FC4	24516 AND R7, R7, R4
			① ▶ x300E	x12A0	4768 ADD R1, R2, #0
			① ▶ x300F	x14E0	5344 ADD R2, R3, #0
			① ▶ x3010	x17E0	6112 ADD R3, R7, #0
			① ▶ x3011	x103F	4159 ADD R0, R0, #-1
			① ▶ x3012	x07F6	2038 BRzp POSE3
			① ▶ x3013	xF025	61477 END1 HALT
			① ▶ x3014	x03FF	1023 NUM .FILL #1023
			① ▶ x3015	x03A2	930 a .FILL #930
			① ▶ x3016	x0001	1 b .FILL #1

- F(2)=2

Registers

R0	xFFFF	65535	
R1	x0001	1	
R2	x0001	1	
R3	x0002	2	
R4	x03FF	1023	
R5	x0000	0	
R6	x0000	0	
R7	x0002	2	
PSR	x8004	32772	CC: N
PC	x3013	12307	
MCR	x0000	0	

Console (click to focus)

```
callback' scheduled for 9938
warning: 10659: Skipping 'Updating Keyboard' scheduled for 10650
warning: 10659: Skipping 'Updating Display' scheduled for 10650
warning: 10659: Skipping 'No interrupt of higher priority pending' scheduled for 10651
warning: 10659: Skipping 'Triggering pre-instruction callback' scheduled for 10658
warning: 12114: Skipping 'Updating Keyboard' scheduled for 12110
warning: 12114: Skipping 'Updating Display' scheduled for 12110
```

Memory

➤ x3000	x5260	21088	AND R1, R1, #0
➤ x3001	x54A0	21664	AND R2, R2, #0
➤ x3002	x56E0	22240	AND R3, R3, #0
➤ x3003	x5920	22816	AND R4, R4, #0
➤ x3004	x5FE0	24544	AND R7, R7, #0
➤ x3005	x16E1	5857	ADD R3, R3, #1
➤ x3006	x5B60	23392	AND R5, R5, #0
➤ x3007	x1B7D	7037	ADD R5, R5, #-3
➤ x3008	x280B	10251	LD R4, NUM
➤ x3009	x1B61	7009	POSE3 ADD R5, R5, #1
➤ x300A	x0C01	3073	BRnz POSE1
➤ x300B	x1241	4673	ADD R1, R1, R1
➤ x300C	x1E43	7747	POSE1 ADD R7, R1, R3
➤ x300D	x5FC4	24516	AND R7, R7, R4
➤ x300E	x12A0	4768	ADD R1, R2, #0
➤ x300F	x14E0	5344	ADD R2, R3, #0
➤ x3010	x17E0	6112	ADD R3, R7, #0
➤ x3011	x103F	4159	ADD R0, R0, #-1
➤ x3012	x07F6	2038	BRzp POSE3
➤ x3013	xF025	61477	END1 HALT
➤ x3014	x03FF	1023	NUM .FILL #1023
➤ x3015	x03A2	930	a .FILL #930
➤ x3016	x0001	1	b .FILL #1

对于测试结果，所测试的都符合要求，过程也符合设计。