第二次实验

姓名: 傅申 学号: PB20000051

1 理论

1.1 尝试寻找通项公式

对于 $F(n) = F(n-1) + 2F(n-3)$, 将递推式化为矩阵形式后如下

$$\begin{pmatrix} F(n) \\ F(n-1) \\ F(n-2) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} F(n-1) \\ F(n-2) \\ F(n-3) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}^{n-2} \begin{pmatrix} F(2) \\ F(1) \\ F(0) \end{pmatrix} \tag{1}$$

然而矩阵的幂的结果十分复杂 (需要计算许多多项式的根), 故不能直接计算通项让 LC3 求解.

1.2 递推

显然, 对于 $F(n)$ 的求解, 可以直接使用递推式, 算法如下:

Algorithm 1: Recurrence

Data: n stored in `R0`
Result: $F(n)$ stored in `R7`

initialization: $\{\texttt{R1}, \texttt{R2}, \texttt{R4}, \texttt{R7}\} \leftarrow \{1, 2, 1023, 1\}$
`/*` `{R2,R1,R7}` `stores` $\{F(n+2), F(n+1), F(n)\}$ `in a row` `*/`
do
  $\texttt{R3} \leftarrow \texttt{R7} \times 2$ `// tmp stores` $F(n-1) \times 2$
  $\texttt{R7} \leftarrow \texttt{R1}$
  $\texttt{R1} \leftarrow \texttt{R2}$
  $\texttt{R2} \leftarrow \texttt{R3} + \texttt{R1}$ `//` $F(n+2) = F(n+1) + 2F(n-1)$
  $\texttt{R0} \leftarrow \texttt{R0} - 1$
while *R0* > 0;
$\texttt{R7} \leftarrow$ `R7 & R4 // R7 & R4` $=$ `R7 mod 1024`
HALT

对于上面的算法的正确性, 有以下两点:

1. 当 n 为 0 时, `R7` 中储存的值为 $F(1)$, 但是由于 $F(1) = F(0) = 1$, 所以算法仍然是正确的.
2. 虽然递推式为$F(n) = (F(n-1) + 2F(n-3)) \bmod 1024$, 但是如果记$G(n) = G(n-1) + 2G(n-3), G(0) = G(1) = G(2) = 1$, 则有 $F(n) = G(n) \bmod 1024$. 在上面的算法中, 虽然可能有溢出情况, 但如果我们将寄存器`R7`中的值$R(n)$看作无符号整数, 则有$R(n) = G(n) \bmod 2^{16}$, 因此$F(n) = R(n) \bmod 1024$, 即算法总能满足取模的要求.

2 L版本程序

2.1 汇编程序

将算法写成汇编程序, 同时, 我的学号为 PB20000051, 即 $a = 20, b = 0, c = 0, d = 51$, 计算得到

$$F(a) = 930, F(b) = 1, F(c) = 1, F(d) = 726$$

将值保存到最后, 有了最终的汇编代码如下, 掐头去尾, 一共 17 行.

```
        .ORIG   x3000
        ADD     R7, R7, #1 ; F(n)
        ADD     R1, R1, #1 ; F(n + 1)
        ADD     R2, R2, #2 ; F(n + 2)
        LD      R4, MOD
LOOP    ADD     R3, R7, R7 ; temp = 2 * F(n - 1)
        ADD     R7, R1, #0
        ADD     R1, R2, #0
        ADD     R2, R3, R1 ; F(n + 2) = F(n + 1) + 2 * F(n - 1)
        ADD     R0, R0, #-1
        BRp     LOOP
        AND     R7, R7, R4
        HALT
MOD     .FILL   #1023
Fa      .FILL   #930
Fb      .FILL   #1
Fc      .FILL   #1
Fd      .FILL   #726
        .END
```

2.2 正确性测试

2.2.1 算法正确性测试

为了测试算法正确性, 我编写了以下的 Python 程序来进行评测.

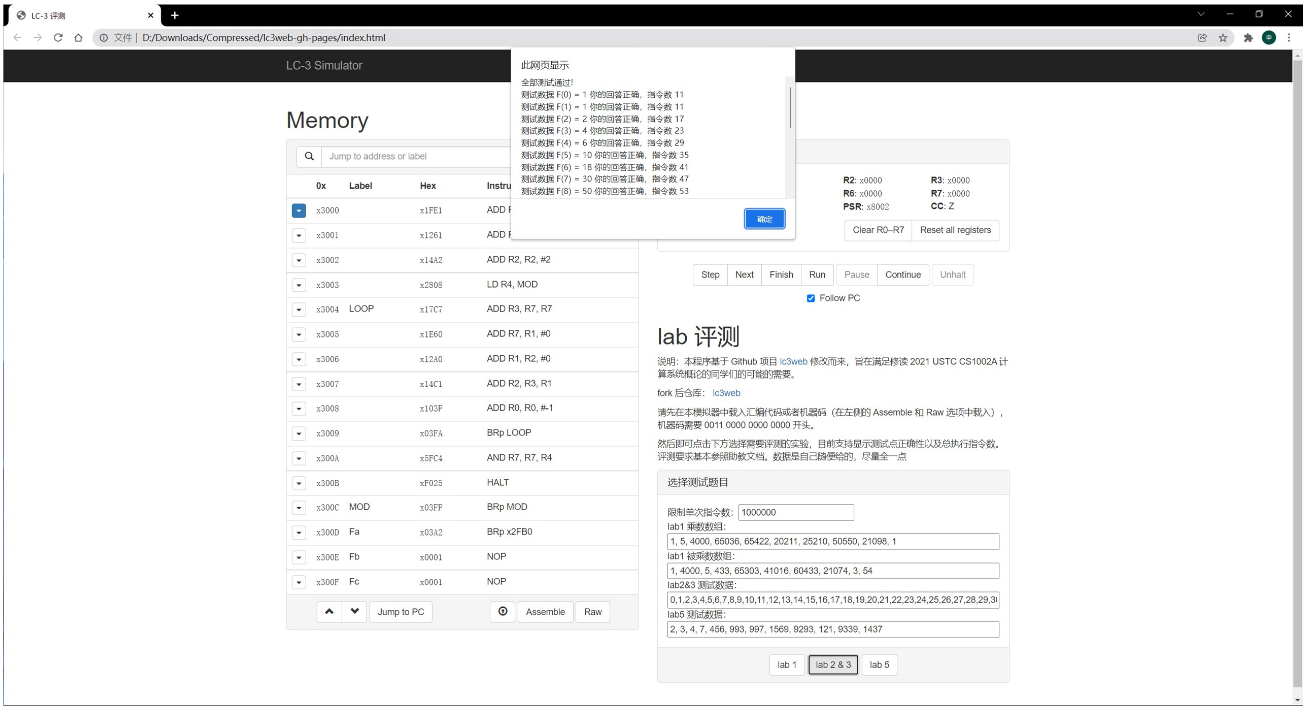
```python
def AlgorithmTest(n):
  r = [n, 1, 2, 0, 1023, 0, 0, 1]
  if n == 0:
    r[0] = 1
  while r[0] > 0:
    r[3] = 2 * r[7]
    r[7] = r[1]
    r[1] = r[2]
    r[2] = r[3] + r[1]
    r[0] -= 1
  r[7] = r[7] & r[4]
  return r[7]

f = [1, 1, 2]
for i in range(3, 16384):
  f.append((f[i - 1] + 2 * f[i - 3]) % 1024)
for i in range(0, 16384):
  if AlgorithmTest(i) != f[i]:
    print("AlgorithmTest(%d) failed," % i,
          "Expected:", f[i], ", Got:", AlgorithmTest(i))
    exit(1)
print("AlgorithmTest passed")
```

```
AlgorithmTest passed
```

可以看到输出结果为 `AlgorithmTest passed` 表示算法正确.

2.2.2 程序正确性测试

使用刘良宇同学的 LC-3 评测网站进行测试, 得到的结果如下图.



可以看到程序测试成功, 结果正确.