ICS Lab Report - lab2

SCGY Cao Gaoxiang PB20000061

December 15, 2021

Lab Name

Better Angels

Lab Purpose

Complete the four tasks. Details are listed below.

Lab Content

1. Read and understand.

   In lab 3, you will get a piece of machine code in 'foo.txt'. Your first task is to translate machine code into assembly code. Store your program in 'translate.txt'.

   However, due to some reasons, assemble code is directly given, so this task is cancelled.

2. Guess.

   This code is some other's program in lab2. Guess the owner of the program by the last 4 lines of the program.

   Write down the owner's student id in 'id.txt'.

3. Optimize.

   The code in lab2 is a L-version program. Of course it's performance is not very well. In this part, you need to optimize other's program. Rewriting is also allowed as an optimization method.

   Store the optimized code in 'optimized.txt'.

4. (optional) Feedback.

   Contact the owner, say anything you like. (Don't swear. If the other party's program is too difficult to understand, please also keep polite.)

Lab Environment

Windows 11 Home Edition version 21H2, Visual Studio Code, LC3Tools v2.0.2.

Lab Procedure

1. As mentioned earlier, this task is cancelled.

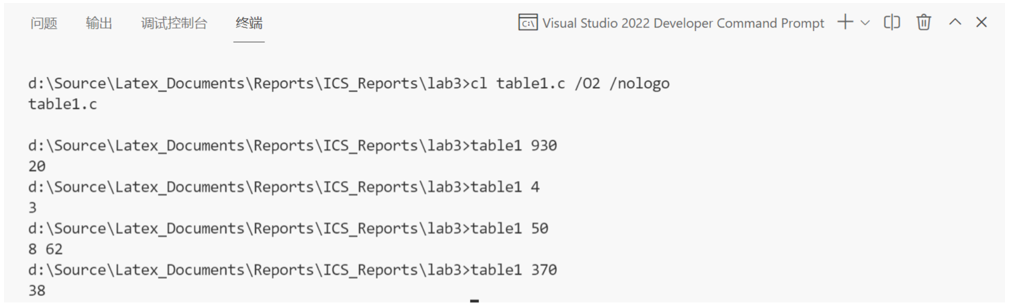
2. The following code is received from get.

```
.ORIG x3000
AND R4, R4, #0
AND R5, R5, #0
AND R7, R7, #0
LD R1, N1
LD R2, N2
LD R3, N3
LD R6, N4
ADD R0, R0, #-2
BRnz Q
N    ADD R4, R3, R1
     ADD R4, R4, R1
BACK    ADD R5, R4, R6
        BRn OK
        ADD R4, R4, R6
        BRp BACK
OK    ADD R1, R2, #0
      ADD R2, R3, #0
      ADD R3, R4, #0
      AND R4, R4, #0
ADD R0, R0, #-1
BRz NEXT
BRnzp N
NEXT    ADD R7, R3, #0
BRnzp #4
Q ADD R0, R0, #2
  ADD R7, R7, R0
  BRp #1
  ADD R7, R7, #1
N1 .FILL #1
N2 .FILL #1
N3 .FILL #2
N4 .FILL #-1024
RESA .FILL #930
RESB .FILL #4
RESC .FILL #50
RESD .FILL #370
.END
```

From the code, it can be concluded that the author of the code uses the same method as me. We both use the algorithm whose time complexity is $O(n)$. Also some information of his student number can be get from the code: $F(a) = 930$, $F(b) = 4$, $F(c) = 50$, $F(d) = 370$. The following code is used to find n when $F(n)$ is given.

```c
#include <stdio.h>
int main(int argc, char const *argv[]) {
    int n = 0;
    if (argc == 1) scanf("%d", &n);
    else sscanf(argv[1], "%d", &n);
    int a0, a1 = 2, a2 = 1, a3 = 1;
    if (n == 1) printf("%d�%d�", 0, 1);
    if (n == 2) printf("%d�", 2);
    for (int i = 3; i <= 100; ++i) {
        a0 = (a1 + 2 * a3) % 1024;
        if (a0 == n) printf("%d�", i);
        a3 = a2; a2 = a1; a1 = a0;
    }
    return 0;
}
```

We get the results as Figure 1. So the author's student number may be PB20030838 or PB20036238. Then we get try to search the numbers in the QQ group, as Figure 2. So

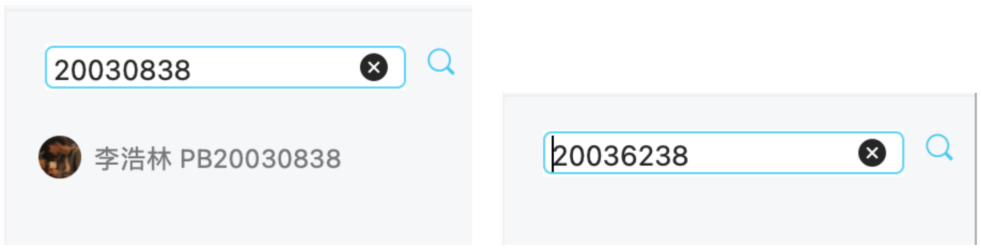


```
d:\Source\Latex_Documents\Reports\ICS_Reports\lab3>cl table1.c /O2 /nologo
table1.c

d:\Source\Latex_Documents\Reports\ICS_Reports\lab3>table1 930
20
d:\Source\Latex_Documents\Reports\ICS_Reports\lab3>table1 4
3
d:\Source\Latex_Documents\Reports\ICS_Reports\lab3>table1 50
8 62
d:\Source\Latex_Documents\Reports\ICS_Reports\lab3>table1 370
38
```

Figure 1: Guess the Student Number

the only possibility is Li Haolin, whose student number is PB20030838.


20030838

李浩林 PB20030838

20036238

(a) PB20030838 (b) PB20036238

Figure 2: Results of the numbers in the QQ Group

3. As mentioned above, we both used an algorithm whose time complexity is $O(n)$. And none of the recursive algorithms can possess a betteer complexity. But since there is modulo operation in the formula, there must be loop in the number queue. After printing all the cases where n is less than 1000, we find out that after 20, $F(n)$ cycles every 128. We can store the 148 cases in the memory, and read them.

   What about going further? Since there are only 16384 cases of input, still smaller than the size of memory of LC-3. (Because our program start at x3000, we have $\frac{13}{16}$ of all the 65536 memory space available. That is 53248, much larger than 16384.) So it is possible so store all the cases in the memory.

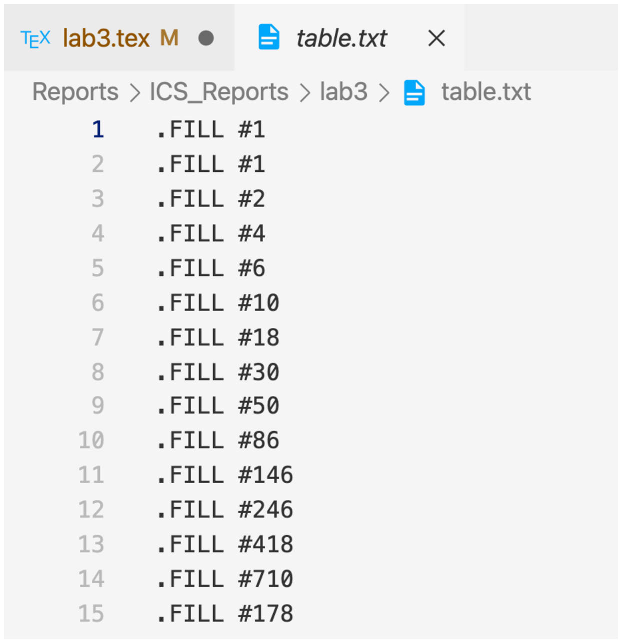
   Because there are too many cases (for human), we use the following program to calculate or the cases, meanwhile store them in assembly format.

```c
#include <stdio.h>
int main(int argc, char const *argv[]) {
    FILE *fout = fopen("table.txt", "w");
    int n = 0;
    if (argc == 1) scanf("%d", &n);
    else sscanf(argv[1], "%d", &n);
    int a0, a1 = 2, a2 = 1, a3 = 1;
    fprintf(fout, ".FILL␣#%d\n", a3);
    if (n >= 1) fprintf(fout, ".FILL␣#%d\n", a2);
    if (n >= 2) fprintf(fout, ".FILL␣#%d\n", a1);
    for (int i = 3; i <= n; ++i) {
        a0 = (a1 + 2 * a3) % 1024;
        fprintf(fout, ".FILL␣#%d\n", a0);
        a3 = a2; a2 = a1; a1 = a0;
    }
    fclose(fout);
    return 0;
}
```

   Results are as Figure 3. (16385 lines)

   Then it's simple to get the answer code as follows. (16391 lines)

```
.ORIG x3000
LD R1, TABLESTART
ADD R1, R0, R1
LDR R7, R1, #0
HALT
TABLESTART .FILL x3004
.FILL #1
.FILL #2
.FILL #4
.FILL #6
.FILL #10
```



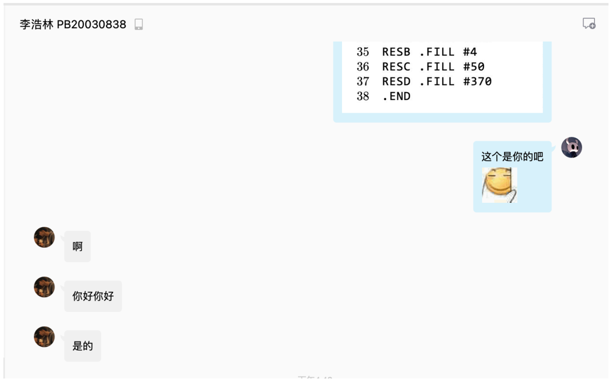
```
.FILL #1
.FILL #1
.FILL #2
.FILL #4
.FILL #6
.FILL #10
.FILL #18
.FILL #30
.FILL #50
.FILL #86
.FILL #146
.FILL #246
.FILL #418
.FILL #710
.FILL #178
```

Figure 3: Table

```
.FILL #18
...
.END
```

No matter what the input is, the program will only perform 3 instructions. Since the data come from C program, we only tested several cases. All passed.

4. We build connection with the author successfully, as Figure 4.


李浩林 PB20030838

```
RESB .FILL #4
RESC .FILL #50
RESD .FILL #370
.END
```

这个是你的吧

啊

你好你好

是的

Figure 4: Build Contact