Lab 2 Report

PB21081601 张芷苒

Purpose

This lab expects us to calculate a variant of the Fibonacci sequence with the following form:

$$F(0) = F(1) = 1$$

$$F(N) = F(N-2)\%p + F(N-1)\%q \ (2 \le N \le 1024)$$

$$p = 2^k \ (2 \le k \le 10), \ 10 \le q \le 1024$$

- p will be stored in x3100, q will be stored in x3101, and N will be stored in x3102
- R0-R7 are set to 0s at the beginning
- the program should start at x3000
- F(N) is expected to be stored in x3103

Principles

For this kind of problem, we usually use the sliding window technique. However in this case, note that $F(N-1)$ and $F(N-2)$ only perform the $\%$ calculation when they are adding, they do not do this while storing.

We can write the pseudo code for this program as follows:

```
Input: p, q, N;
Output: F(N);
n1 = 1;
n2 = 1;
N = N - 1; //计数器
while (N > 0) {
    temp1 = n1 % p;
    temp2 = n2 % q;
    F = temp1 + temp2;
    n1 = n2;
    n2 = F;
    N = n - 1;
}
```

Procedure

register distribution

We need 8 registers to store the 8 variables above.

the $\%$ operation

$\%$ is done by subtracting R2 from R1 until the result is negative, and then add R2 once.

the minus operation

Using the ADD instruction, $R1 - R2 = R1 + (R1)_{reverse} + 1$.

calculating the Fibonacci sequence

For a regular Fibonacci sequence $F(N) = F(N-1) + F(N-2)$, assume there is a window the length of 3. Every time the sequence increases, the window move one position to the right on the sequence, until the increment ends. The latter window stores the result of the former window.

the code

1. initializing

$$R_0 \leftarrow p, R_1 \leftarrow q, R_2 \leftarrow N$$

```
LD R0, x0FF ;R0 <- p
LD R1, x0FF ;R1 <- q
LD R2, x0FF ;R2 <- N
```

2. prepare minus

$$R_3 \leftarrow -p, R_4 \leftarrow -q$$

```
NOT R3, R0     ;reverse
ADD R3, R3, x1 ;plus 1
NOT R4, R1
ADD R4, R4, x1
```

3. $\%$

$$R_5 \leftarrow F(N-2)\%p, R_7 \leftarrow F(N-1)\%q$$

```
ADD R5, R5, R3
BRzp #-2
ADD R5, R5, R0 ;F(N-2) % p
ADD R7, R6, #0
ADD R7, R7, R4
BRzp #-2
ADD R7, R7, R1 ;F(N-1) % q
```

4. add

$$R_7 \leftarrow F(N-2)\%p + F(N-1)\%q$$

```
ADD R7, R7, R5 ;R7 <- F(N-2) % p + F(N-1) % q
```

5. move

$$R5 \leftarrow R_6, R_6 \leftarrow R_7$$

```
ADD R5, R6, #0 ;R5 <- R6
ADD R6, R7, #0 ;R6 <- R7
```

6. store the result

$$R_7 \rightarrow x3103$$

```
ST R7, x0EC ;R7 -> x3103
```

Question

> *answer the question: How can you improve the efficiency of the loop structure in your program? (Just describe your idea briefly.)*

Since p is the power of 2, in binary form p can be represented by and only by a number of 1s. Therefore the mod calculation $F(N-2)\%p$ can be simplified by using the AND instruction, instead of doing the subtraction over and over again. This improves the program efficiency significantly.

```
ADD R3, R0, #-1
AND R5, R5, R3
```

Results

Test results:

汇编评测

3 / 3 个通过测试用例

- 平均指令数: 2248.6666666666665
- 通过 256:123:100, 指令数: 1182, 输出: 146
- 通过 512:456:200, 指令数: 2196, 输出: 818
- 通过 1024:789:300, 指令数: 3368, 输出: 1219