ICS 2021　LAB06

实验报告

PB20111699　吴骏东

2022.1.3

一．任务概述

本次实验要求使用高级语言对先前的所有实验进行重构。要求核心算法不能进行改变，即与自己先前的代码执行模式保持高度的一致性。

二．实现

为了模拟 LC3 汇编指令的执行方式，我采用了如下的策略。

1. 用全局变量模拟寄存器。16bit 的大小恰好等同于 c++的短整型变量大小。故以此替代。使用全局变量的好处是可以在程序的各个部分对其进行修改，这与 LC3 的模式是一样的。寄存器的特定初值使用 scanf 读入。最终的指令框架如下：

```
#include <cstdio>

short r0 = 0, r1 = 0, r2 = 0, r3 = 0, r4 = 0, r5 = 0, r6 = 0, r7 = 0;
//初始化寄存器
int main() {
	scanf("%hd", &r0);
	//begin
	……
	//end
	printf("%hd", r7);
	return 0;
}
```

不同的程序只需要修改 begin 至 end 之间的相应内容即可。

2. 用全局变量模拟.fill 指令。LC3 中的.fill 会开辟一个专门的空间用来存储数值，这与 c++的变量模式相同。在本次实验中，为了规范变量使用，如

果.fill 的空间不涉及值修改操作，则使用 const short 型变量进行替代；若涉及值修改操作则使用 short 型变量替代。

3. 用函数模拟 JMP & RET。LC3 的 JMP 与 RET 实现了 PC 在不同代码部分之间的跳转，从而让程序功能变为一个整体进行调用。该操作与 c++中的函数思想高度一致。因此我选择使用无输入值、无返回值的函数模拟这一部分。JMP 对应函数调用，RET 对应函数内 return，从而保证了功能的整体性。

4. 用 if-goto 模拟 BR 指令。BR 指令的功能是根据 nzp 的值进行 PC 跳转。nzp 值可以通过 if 指令进行判断，PC 跳转可以通过 goto 指令实现。该操作虽然违反了 c++编程规范，但可以完美契合 LC3 指令的运行逻辑。

5. 保留了 add、and 指令。使用 c++进行重构时，代码中只出现了+、&运算符。此外，为了化简代码内容，我使用了-运算符替代原先的取反加一。这样保证了重构过程的一致性与契合性。

以上是本实验使用 c++重构 LC3 汇编代码的基本逻辑。不同程序中的相关细节处理见相应源码文档。

三. 小结

使用高级语言进行重构，我们发现高级语言的逻辑十分清晰，符合人类的思维模式与阅读模式。LC3 因为其简洁性而没有加入许多功能性指令，例如乘法、移位、按位或等。事实上如果 LC3 汇编指令中添加了移位运算符，程序的复杂度会显著下降。但是 LC3 作为基础的底层语言，对我们理解程序的运行逻辑、代码效度等有着许多帮助。