第五次实验报告

201250044 王星云

一、程序实现功能、实现方式、精巧设计

（一）程序实现功能

程序将使用SysY语言书写的源代码翻译为LLVM IR，本次实验翻译

1. 函数定义与函数调用的翻译
    - 函数定义的参数不为数组，但调用时可能会有将数组某一个元素作为实参传入的情况
2. 局部变量的声明，定义，使用
    - 除普通的int类型变量外还需要翻译常量与数组变量。
    - 数组仅为一维数组
    - 变量也会参与表达式运算

（二）实现方式

1. 设计符号表
    - 设计一个符号表，存储变量（函数以及局部变量），变量属性包括LLVMValueRef。
2. 翻译局部变量存取
    - 在定义const常量、变量、函数参数时，为变量分配空间、存储变量、将变量的名称以及其ValueRef存储到符号表。
    - 在使用变量时，通过变量名从符号表获得变量及其ValueRef，之后加载变量、使用变量。
3. 翻译数组存取
    - 数组是一种特殊变量，在定义和使用时，需要使用GEP指令
4. 翻译函数定义和使用
    - 需要注明的是，返回值为void类型的函数，在调用时没有返回值
5. 翻译return语句
    - 这部分内容大致与lab4相同。需要注明的是，返回值为void类型的函数，return的是void。此外，即使源码中函数没有返回值，在IR中也要加上返回值。

（三）精巧设计

在函数定义时，通过判断函数子节点是否有return语句的实例，来判断源码中函数是否有return语句——如果没有，那么在函数定义时根据函数类型生成return语句。

二、有趣的现象和印象深刻的bug

（一）有趣的现象

尽管我自己生成的中间代码和clang生成的看起来很不一样，但是使用`lli`命令后的执行结果都是相同的（如果不同，那就是我的代码出现问题了！）。

（二）印象深刻的bug

1. 即使源码中函数没有返回值，在IR中也要有return语句

   比如:

   ```
   1 void test(){}
   ```

   仍然需要生成中间代码 `ret void`

2. 我虽然翻译了函数调用语句，但是没有处理 `(exp)` 类型的语句，导致有一个用例一直报错 `core dumped`