

## 《编译原理》实验 4 目标代码生成

姓名：缪源清 学号：161220093 邮箱：myqlily@126.com

### 1.实验任务

将生成的中间代码翻译成目标代码。

### 2.实验平台

windows10 + Vmware + ubuntu16.04

### 3.实现思路

#### (1) 指令选择机制

复用实验三中间代码的生成框架，根据指令的类型进行翻译。

#### (2) 寄存器分配算法

采用局部寄存器分配算法：

##### 1) 基本块划分：

函数头、IF、call、goto、label 作为划分基本块的准线。在这些位置，修改过的变量需要被重新装入内存。

##### 2) 寄存器选择：

若当前变量已经在某个寄存器中，则返回该寄存器的编号。若当前变量不在，如果此时有空闲的寄存器，则分给它一个空闲的寄存器；如果没有空闲的寄存器，则选择一个寄存器，将其中的数据装入内存。选择算法是：选择最久未被使用的寄存器。

#### (3) 堆栈管理

使用\$fp 定位。因此堆栈的形式为

	parameters		<- \$fp
	ret addr		
	old fp		
	data		
			<- \$sp

### 4.数据结构

在寄存器的分配中，使用到了变量描述符和寄存器描述符，它们的结构为：

```
typedef struct VarDes_t{
    Operand op;
    int regNo;
    int offset;
    struct VarDes_t* next;
} VarDes;

typedef struct RegDes_t{
    char name[3];
    VarDes* var;
    int old;
} RegDes;
```

目标代码是适用于 MIPS 的, 本实验将 t0~t9 和 s0~s7 共 18 个寄存器作为寄存器分配中的“可用寄存器”。RegDes\_t 的 name 字段记录的是该寄存器的别名(如 t1), var 字段指向当前存放的变量, old 字段记录了此寄存器上一次被引用的时间, 是寄存器选择算法中重要指标。

## 5.编译方法

在/Code 下

make clean: 删除生成的文件

make: 编译、生成 parser

make t1: 用 parser 编译/Test/t1.cmm (样例 1), 并运行 spim

make t2: 用 parser 编译/Test/t2.cmm (样例 2), 并运行 spim

若测试某新文件:

./parser 待测试文件名 生成的文件名 (以.s 结尾)

spim -file 生成的文件名 (以.s 结尾)

## 6.实验结果

样例 1:

```
SPIM Version 8.0 of January 8, 2010
Copyright 1990-2010, James R. Larus.
All Rights Reserved.
See the file README for a full copyright notice.
Loaded: /usr/lib/spim/exceptions.s
Enter an integer:7
1
1
2
3
5
8
13
```

样例 2:

```
SPIM Version 8.0 of January 8, 2010
Copyright 1990-2010, James R. Larus.
All Rights Reserved.
See the file README for a full copyright notice.
Loaded: /usr/lib/spim/exceptions.s
Enter an integer:7
5040
```

## 7.实验不足

不能支持含有数组的运算, 除此之外满足指导书上的各项假设。

## 8.参考

实验 4 的寄存器分配算法参考了 <https://github.com/BlodSide/Principles-and-Techniques-of-Compiler/blob/master/lab4/objCode.c>, 特此说明。

## 9.bug 修复

修复了实验三中有关分支的 bug。