# 实验四 目标代码生成

许嘉禾 191220135

## 一、实现功能

必做内容。

## 二、如何编译

用给定的 Makefile 即可。

## 三、具体实现

首先为了方便基本块的划分，将之前用双向循环链表存储的中间代码转换为数组，这样可以用下标表示基本块的首尾。

对于整个程序，先按函数进行划分，再在每个函数内部进行基本块划分。因为不同函数之间的变量都是完全不同的，且每个函数有独立的栈帧，所以先按函数划分会带来很多方便。

如手册所说，目标代码生成主要是三项工作：指令选择、寄存器分配、栈的管理和过程调用的处理。指令选择直接利用手册上提供的中间代码到目标代码的对应表即可，主要是要在生成指令前完成寄存器的分配。我才用的是手册上的第二种办法，即有空闲就把空闲寄存器直接给它，没空闲的就找到其中保存的变量在接下来不会出现或最晚出现的那个寄存器，然后将里面原来保存的变量溢出到栈中，同时修改寄存器信息和变量位置信息。需要维护两个数据结构，一个是寄存器中保存的变量，一个是各个变量当前所在位置（寄存器编号或栈中的偏移）。

对于栈的管理，我在每个函数开始前首先划分一块区域（也就是把 sp 寄存器的值减去一个值），函数返回前再把 sp 值加回去（在 jr 指令前）。具体分配的大小，我是先遍历该函数，统计数组需要的大小，再在所有数组所需空间的基础上再加 10000（即假设各种寄存器中保存不下的变量加起来不超过这个数）。每个函数都有一个 offset 变量保存当前可以在栈中存放的位置，主要是数组地址和从寄存器中溢出的变量的地址。在函数调用的时候，也要先将所有寄存器中的变量溢出到内存，在 jal 指令之后再全部还原。这里我是将 t0-t9 以及 s0-s7 这 18 个寄存器都看成等价的，都在处理 call 指令的时候由调用者保存，没有遵循规范，只是自己写起来方便。