**代码生成实验报告**

班级：07111505

学号：1120151839

姓名：徐宇恒

**一、实验目的**

根据所学的编译原理理论知识，利用BIT-MiniCC框架，将语法分析器生成的xml文件形式的语法分析树作为输入，最终生成MIPS汇编代码。

**二、实验内容**

将Java语言作为实现语言，在语法分析的基础上，选择MIPS汇编语言作为最终的目标语言进行代码生成。根据汇编语言编写规则，结合寄存器中数值与语法树结构，生成目标代码。

**三、实验过程**

1、语法树变换为四元式

语法树是代码生成过程中的一种中间表示，语法树的叶子代表寄存器、存储单元或常量值，内部结点代表在操作数上进行的操作。

文法经过词法分析与语法分析后形成语法树，并进行调整，得到对应的四元式。

2、寄存器分配

MIPS的体系结构含有丰富的指令系统，并且他是载入/存储模式的体系结构，即所有的指令必须使用寄存器或立即数作为指令的操作数，不允许使用存储器。结合MIPS体系结构的特点，对寄存器进行分配使用。

寄存器操作比内存操作代码短，速度快。在BIT-MINICC框架中，寄存器分配阶段主要考虑的是$a0-$a3，$t0-$t9，$s0-$s7，这些寄存器中保存的内容如下表所示：

|  |  |
| --- | --- |
| **命名** | **存放内容** |
| $a0-$a3 | 过程参数 |
| $t0-$t9 | 临时变量 |
| $s0-$s7 | 保存变量 |
| $gp | 全局指针 |
| $sp | 堆栈指针 |
| $ra | 过程返回地址 |

另外，在MIPS体系结构中，所有的指令都是32位，任何指令都不可能只占用2或3个字节的内存空间。在MIPS中，定义了三种指令格式：R-类型，I-类型和J-类型。其中R-类型指令对三个寄存器操作，I-类型指令对两个寄存器和一个16位立即数操作，J-类型指令对一个26位的立即数操作。

以指令add，a，b，c为例分析寄存器分配，指令操作为a=b+c，b、c存放于高位，a存放于低位，首先将变量b的地址存入寄存器$t0中，将变量c的地址存入寄存器$t1中，进行加法运算后，将结果存入寄存器$t0中。

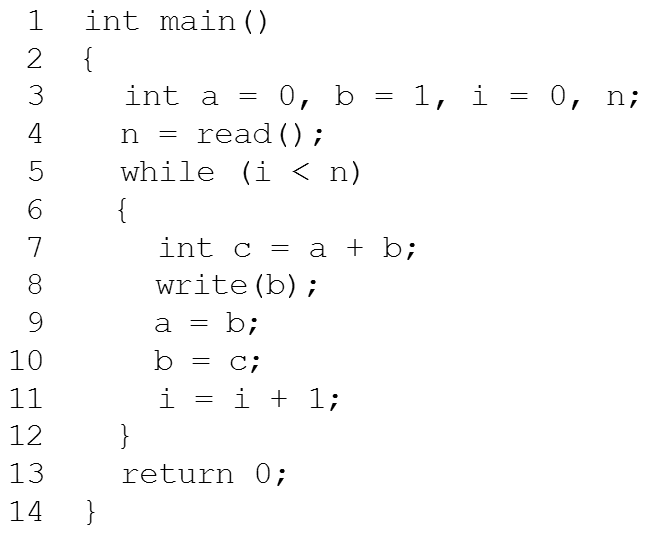
3、目标代码生成

根据语法树中结点的信息与寄存器中数值生成最终目标代码，各语句的语义与相对应的汇编代码规则如下：

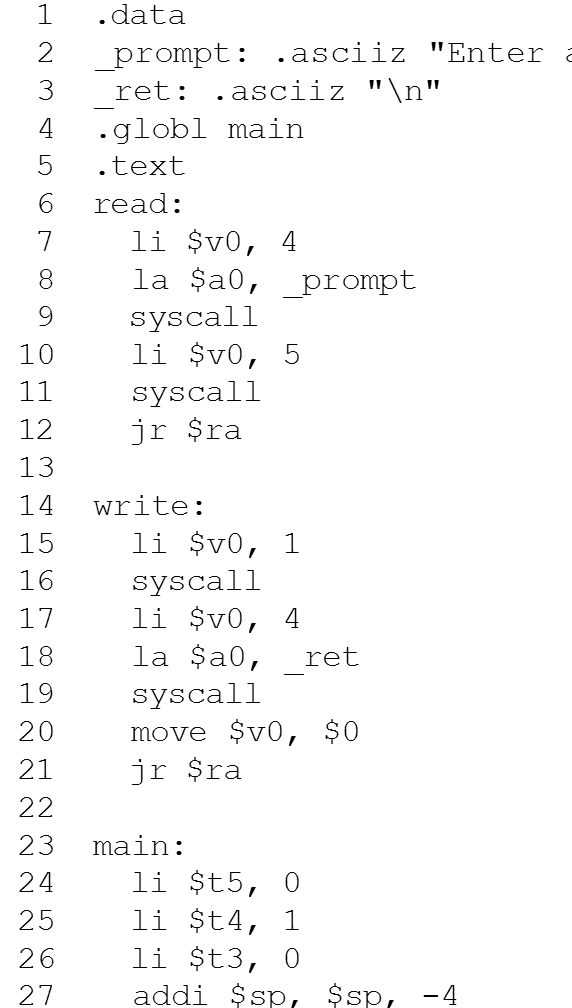
|  |  |  |
| --- | --- | --- |
| **指令功能** | **汇编语句示例** | **代码语义** |
| 取立即数 | LI $S1，100 | $S1=100 |
| 加 | ADD S1，S2，S3 | $S1=$S2+$S3 |
| 减 | SUB S1，S2，S3 | $S1=$S2-$S3 |
| 乘 | MUL S1，S2，S3 | $S1=$S2\*$S3 |
| 除 | DIV S1，S2，S3 | $S1=$S2/$S3 |
| 取字 | LW S1，100(S2) | $S1=MEMORY[$S2+100] |
| 存字 | SW S1，100(S2) | MEMORY[S2+100]=S1 |
| 条件语句 | BEQ S1，S2，100 | IF(S1==S2) PC+4+400 |
| 跳转指令 | JMP 2000 | GOTO 8000 |

**四、实验结果**

测试程序：



实验结果：



**五、实验心得**

经过一学期的编译原理课程的学习，我真正体会到了要实现编程语言的编译有多难，从最开始的词法分析到之后的语法分析，还有最后这一次目标代码的生成，我都经历了一个从一筹莫展，然后慢慢摸索最后写出了一点能够符合要求的代码。最后的目标代码生成和前面几次实验相比，这个实验难度更大，所能找到的资料更少，所以我做的完成度也不高。但是正如老师最开始说的一样，编译原理确实是一门难度很大的课程，除了平时上课的听讲，课后的及时复习之外，还要课下好好做实验，亲自操作以后才能更好的掌握。