**编译原理课程设计实验报告**

实验名称： 基于BIT\_MiniCC框架的语义分析及代码生成器的开发

姓名/学号： 1120141947 郝程乾

1. **实验目的**

通过实践深入理解语法分析的过程，并对JAVA环境下的开发进行了解。

1. **实验内容**

**语义分析阶段的工作主要包括两部分**:

(1)基于词法分析获得的分析树，构建符号表，并进行语义检查。如果存 在非法的结果，请将结果报告给用户;

(2)将分析树转换为抽象语法树并输出。

**代码生成阶段的工作部分：**

基于 BIT-MiniCC 构建 C 语法子集的代码生成 模块，该语法分析器能够读入 XML 文件形式的语法分析树，进行寄存器分配， 并生成 MIPS 或者 X86 汇编代码。

生成汇编程序是该部分的基本要求;进一步地，如果生成的是 MIPS 汇编，则要求汇编代码能够在 BIT-MiniCC 集成的 MIPS 模拟器中运行(注意config.xml 的最后一个阶段“simulating”的“skip”属性配置为“false”);如 果生成的是 X86 汇编，则要求使用 X86 汇编器生成 exe 文件并运行。

1. **实验环境**

操作系统：Windows 10 专业版。

Java运行环境：Java(TM) SE Runtime Environment (build 1.8.0\_111-b14)

Java Hot Spot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode)

集成开发环境：Eclipse Mars.2 Release 4.6.2

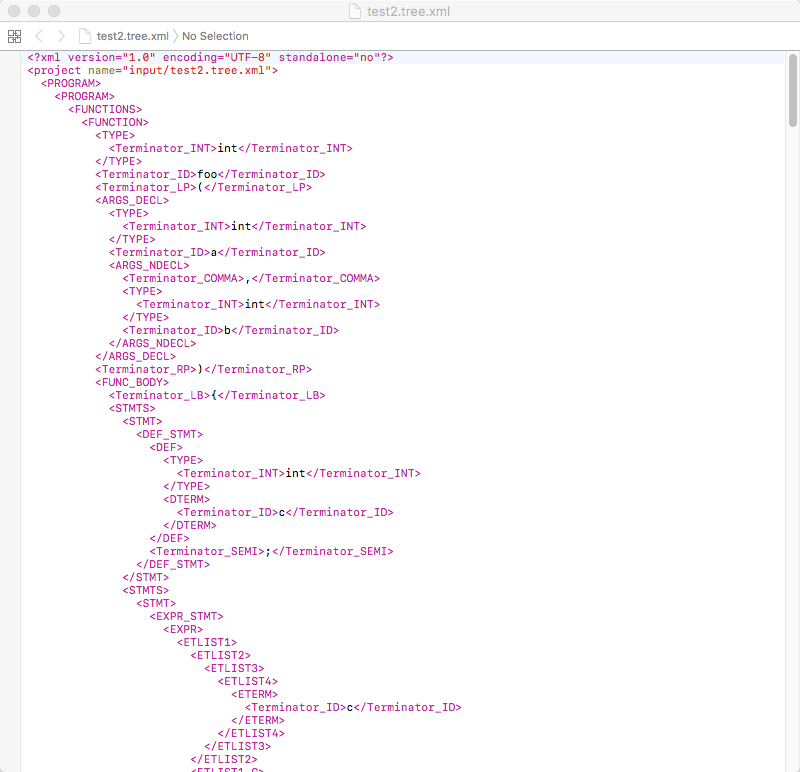
1. **程序设计与实现**

本次实验采用了递归向下分析法对已经生成并抽象的语法树进行分析，最终输出MIPS汇编代码。

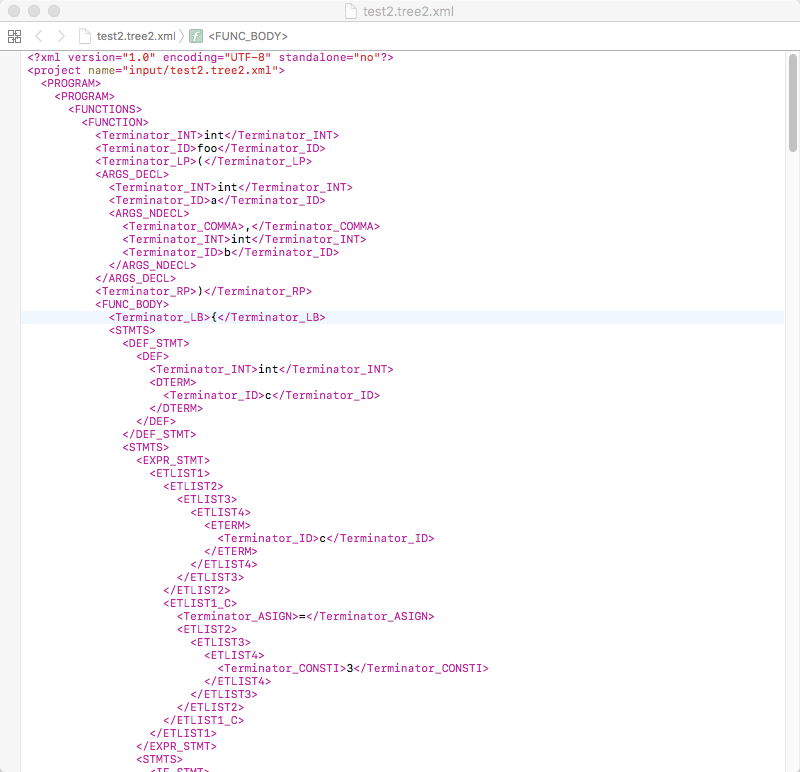
4.1 语法树的抽象

由于语法分析结束后的语法树过于庞大冗杂，需要对其进行进一步抽象以达到削减递归向下分析法步骤的目的。

所以先通过后续遍历的方式找到所有可以优化的节点与分支，然后再通过前序遍历的方式重构整个语法树，剔除所有无用的节点和分支，并对语法树进行输出对比。



图一 平衡前的语法树



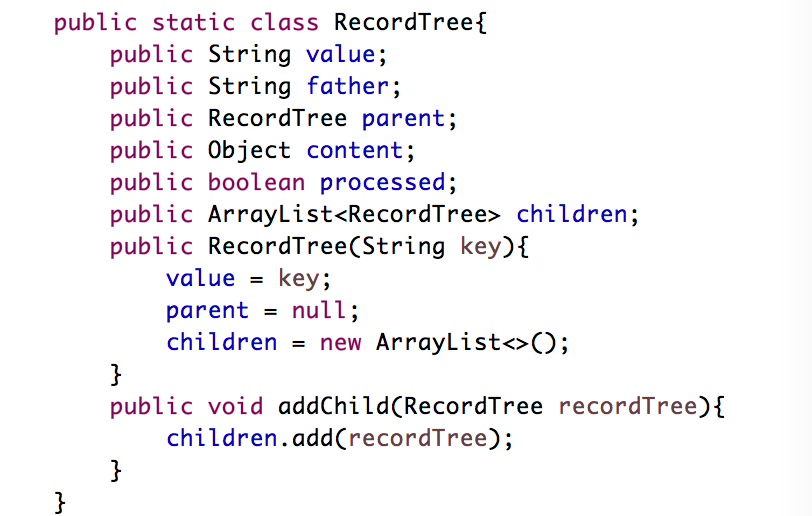
图二 平衡后的语法树

很明显可以看出，当前的语法树中无用的符号节点，以及无用的非直接属性节点均已经被剔除。保留的节点均为属性节点或者分支节点。

至此，语义分析的初始工作已经准备就绪。

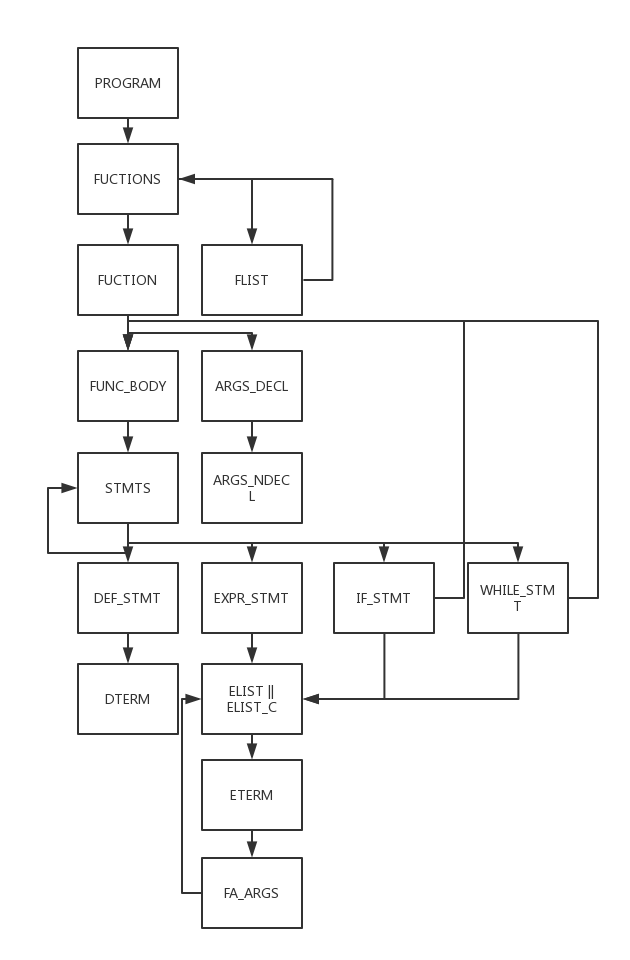
4.2语法树的递归下降分析

由于语法树已经抽象完毕，所以所有节点的属性与所处文法产生式的位置均已经一目了然。

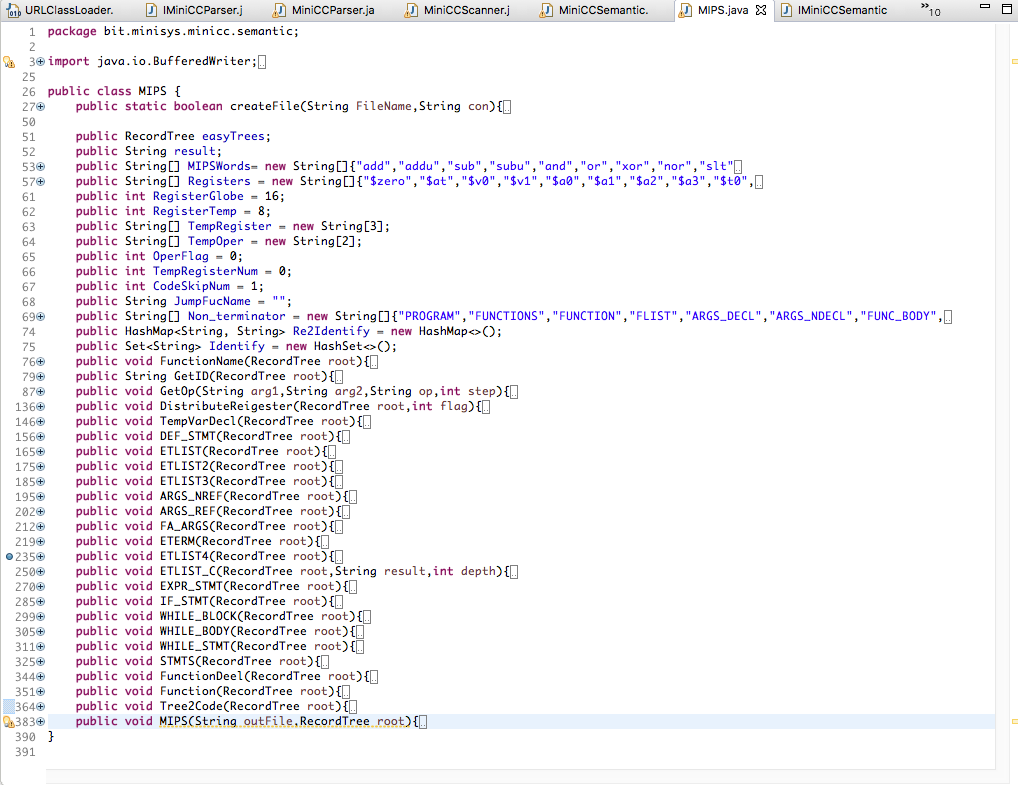


图三 语法树节点的定义

按照语法树节点的定义方式，通过递归下降分析法，并根据如图四的层次分析模型，对语法树进行分析。



图四 语法树递归函数模型



图五 代码函数表

如图五，代码层次结构与语法树递归函数模型相一致。

4.3 代码生成

由于递归下降分析时可以借助每一个终结符号的属性来推断当前产生式所代表的语义是什么。根据已知的语义，可以构造出响应的MIPS代码模板，比如：

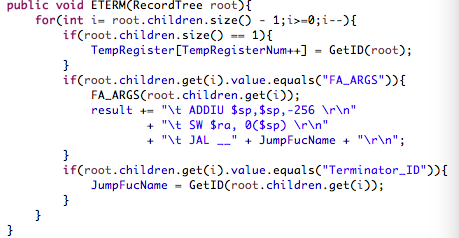
对寄存器的分配基于当前语义的分析，当递归下降分析器分析到了DEF\_STMT，就表明这个产生式内的所有属性为Terminal\_ID的终结符号均应当被分配寄存器。而分配寄存器则是根据MIPS寄存器的分配逻辑进行分配。分配完成后，输出分配代码即可。

对赋值语句的代码生成相对复杂一些，由于本程序最多支持三个操作数两个运算符的等式，所以相对而言，只需要对两种情况进行判断并输出代码即可。可是，本程序支持11种运算符，所以在运算符代码的生成上，情况还是比较复杂的。从图六可以看出，对条件跳转语句的判断是一个难点，因为条件跳转需要考虑到跳转和不跳转的代码块的声明，所以根据递归调用的逻辑，对代码块顺次声明即可。



图六 表达式代码生成函数

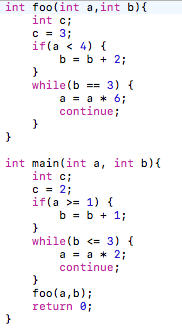
对于函数调用来说，需要对函数运行的栈进行清空和处理。由于本程序只考虑一次函数调用，不考虑嵌套调用，所以对栈的处理以及返回入口寄存器可以将其MIPS代码写死如图七。



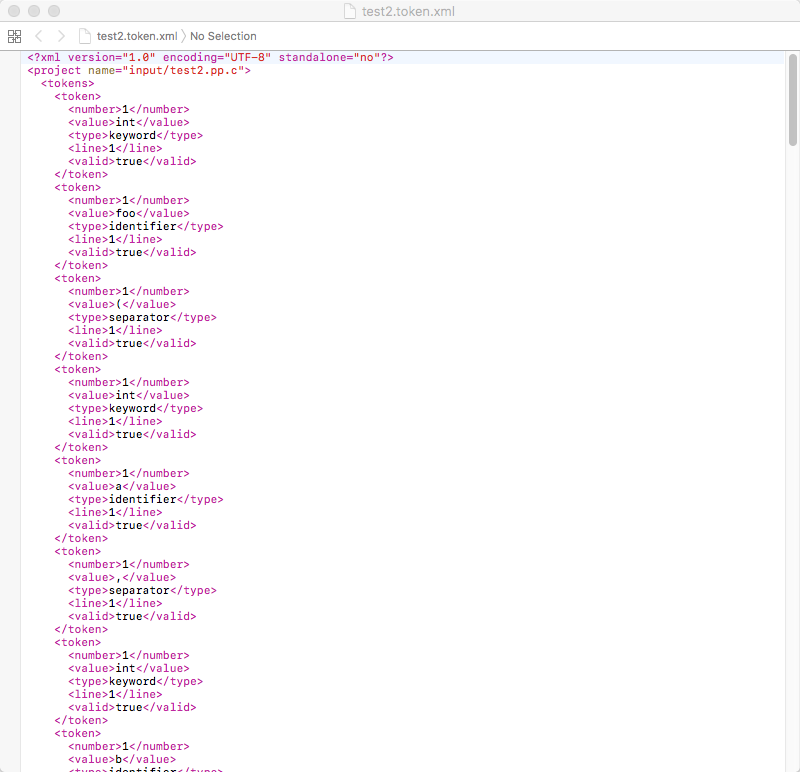
图七 表达式代码生成函数

4.4语义分析与代码生成的结果展示

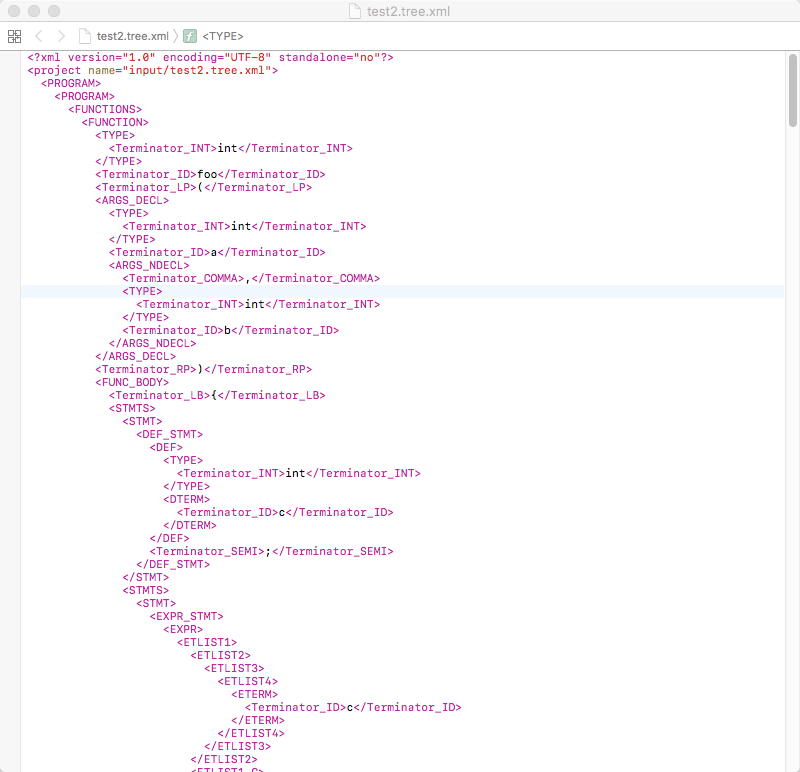
4.4.1 原程序图：



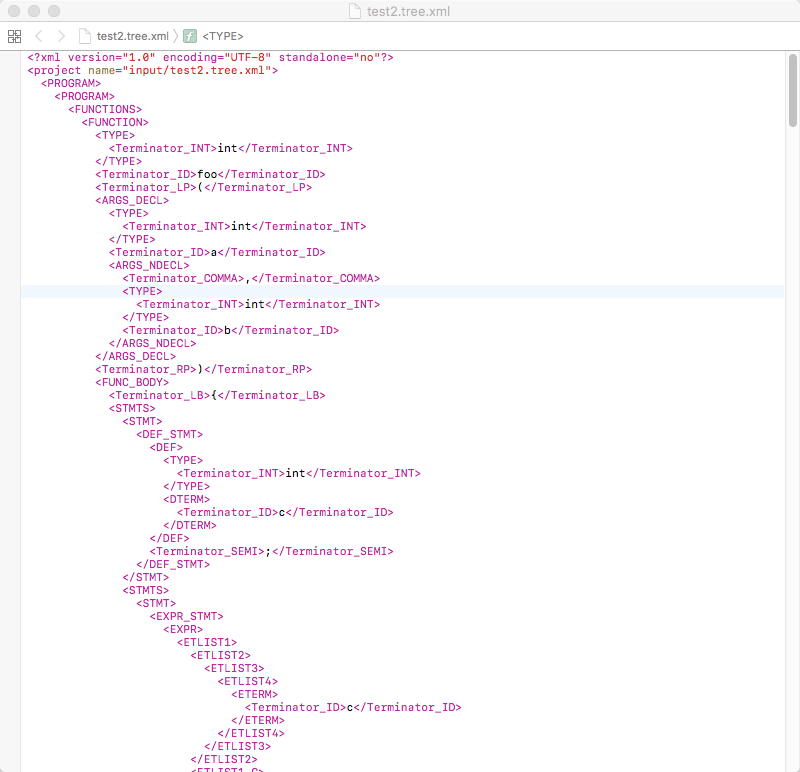
4.5.2 词法分析后部分结果：



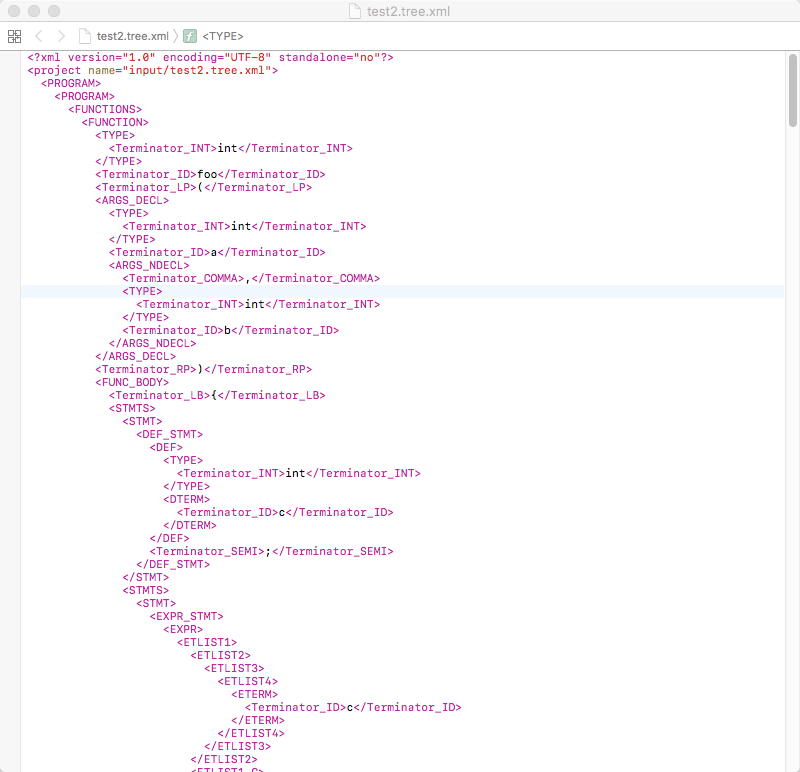
4.5.3 语法分析后部分结果：



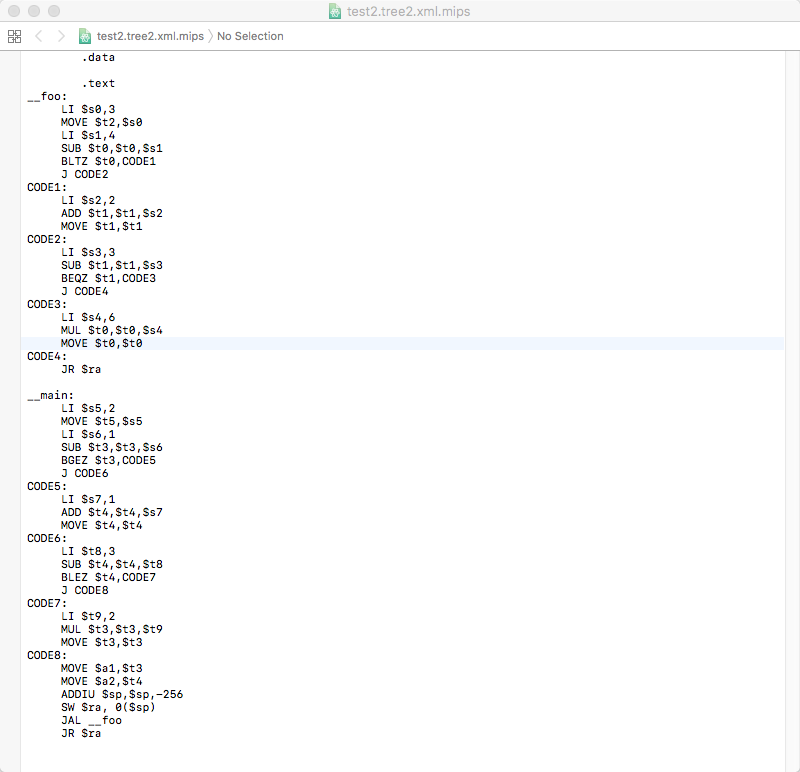
4.5.4 语义分析后部分结果：



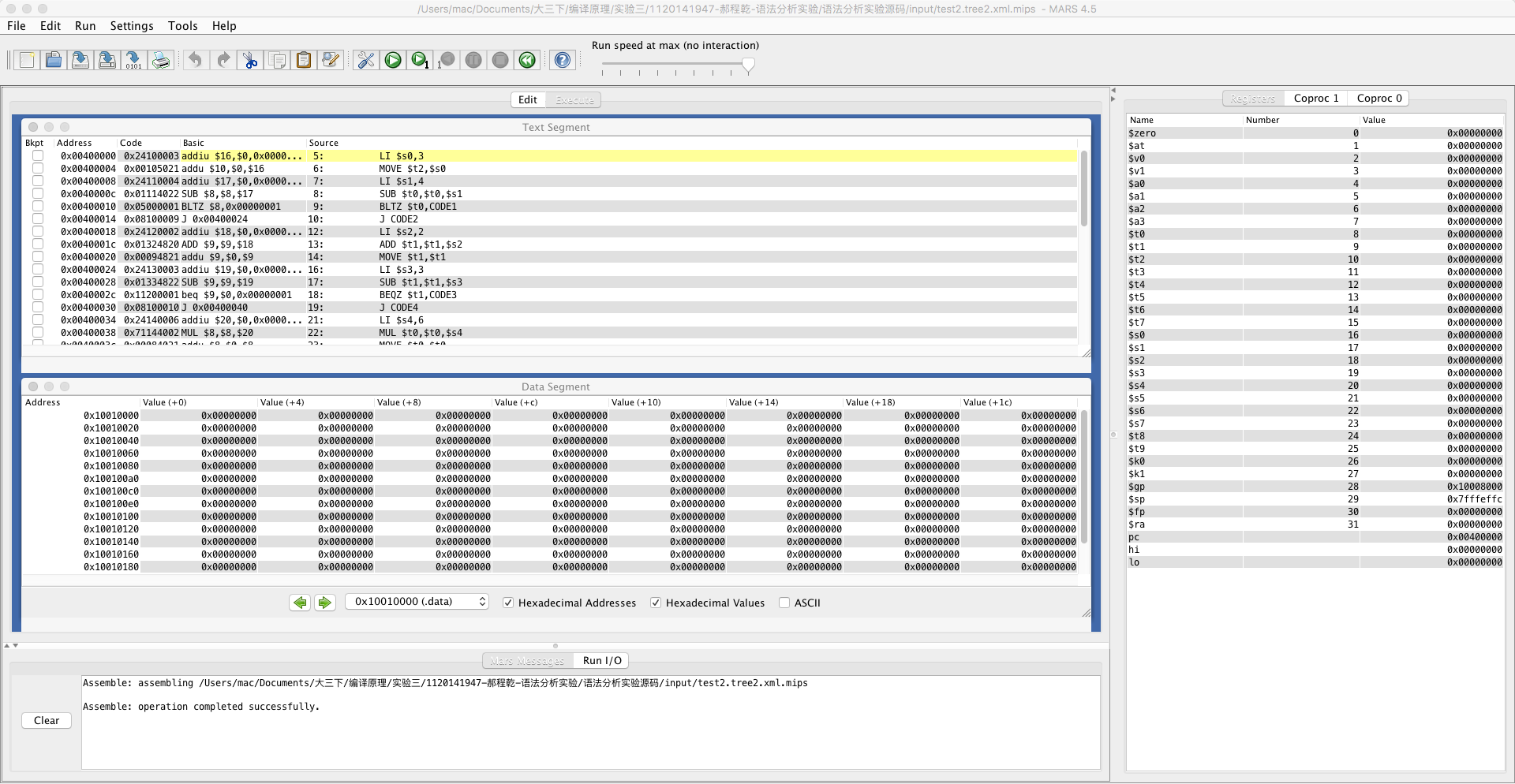
4.5.5 语义分析后部分结果：



4.5.6 代码生成后部分结果：



4.5.7 MIPS编译器语法筛查后结果：



可以通过MIPS语法筛查，实验成功。

**实验收获与体会**

本实验的目的在于缩短理论与实践、前端与后端的距离。

通过对语义分析和代码生成的编译器功能完全实现，不仅提高了我的变成水平，更加深了我对编译原理这门课的理解。让我牢固地掌握了编译器对语言的分析方法，收获非常大。

由于我使用的是递归向下语义分析与LL（1）语法分析，两者在编程逻辑上的完全不同，给我带来了很大的挑战，但是通过耐心地编写代码与Debug成功实现了所有曾经想实现的功能，非常有成就感。