重庆大学编译原理课程实验报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **年级、专业、班级** | | **20计科06班** | | **姓名** | **方成林** |
| **实验题目** | **编译器设计与实现** | | | | |
| **实验时间** | **2023/6/10** | | **实验地点** | **DS3304** | |
| **实验成绩** |  | | **实验性质** | **□验证性 □设计性 ■综合性** | |
| 教师评价：  □算法/实验过程正确；□源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  其他：  评价教师签名： | | | | | |
| 一、实验目的  以系统能力提升为目标，通过实验逐步构建一个将类C语言翻译至汇编的编译器，最终生成的汇编代码通过GCC的汇编器转化为二进制可执行文件，并在物理机或模拟器上运行。实验内容还包含编译优化部分，帮助深入理解计算机体系结构、掌握性能调优技巧，并培养系统级思维和优化能力。 | | | | | |
| 二、实验项目内容  本次实验将实现一个由 SysY (精简版 C 语言，来自 [https://compiler.educg.net/](https://compiler.educg.net/" \t "http://114.117.246.238:4000/_blank)) 翻译至 RISC-V 汇编的编译器，生成的汇编通过 GCC 的汇编器翻译至二进制，最终运行在模拟器 qemu-riscv 上  实验至少包含四个部分: 词法和语法分析、语义分析和中间代码生成、以及目标代码生成，每个部分都依赖前一个部分的结果，逐步构建一个完整编译器  ****实验一****：词法分析和语法分析，将读取源文件中代码并进行分析，输出一颗语法树  ****实验二****：接受一颗语法树，进行语义分析、中间代码生成，输出中间表示 IR (Intermediate Representation)  ****实验三****：根据 IR 翻译成为汇编  ****实验四(可选)****：IR 和汇编层面的优化 | | | | | |
| 1. 实验内容实现   请回答：   1. 实现哪些内容 2. IR库的使用，如何使用静态库链接，如何使用源代码来构建库？结合CMakelist说明 3. 在IR中你如何处理全局变量的，这样的设计在后端有什么好处？后端中如何处理全局变量？ 4. 在函数调用的过程中，IR测评机发生了什么？ 5. 如何处理数组作为参数的情况，为什么可以这样做？ 6. 如何支持短路运算？ 7. 是否进行了寄存器分配，使用了什么方法？ 8. 在函数调用的过程中，汇编需要如何实现，汇编层次下是怎么控制参数传递的？是怎么操作栈指针的？ 9. 是否进行IR或者后端的优化，是如何实现的？   评分标准：必须回答1，2-9中选择3个或以上问题进行回答，至少3个缺一项扣2分  1实现内容：词法分析，语法分析，语义分析，中间代码，目标代码生成。  3在analysisCompUnit函数中，每次遇到变量声明则说明是全局变量，首先在声明这个变量时，在def阶段，可以获取到对应的变量名以及值，所以在这个函数中将对应的语句加入到global函数中。在get\_ir\_program函数中，当处理完所有代码后，遍历global作用域的符号表，将所有的全局变量加入到globalVal中去。  5在处理函数参数时，判断root的字节点数目，如果大于2则说明是数组，获取到数组的长度后，将变量存储进dimension中。  8函数参数通过寄存器传递，首先将栈指针压栈，然后将寄存器中的值存储进栈中，在函数返回后恢复到之前的位置。 | | | | | |
| 四、实验测试  请回答：   1. 测试程序是如何运行的？执行了什么命令？你的汇编是如何变成RISV程序并被执行的？   IR测评机执行ir::program，使用int run（）函数从main开始执行整个program并且返回main的返回值，IR 测评机会根据 GlobalVal 的 maxlen 字段和本身的类型为该全局变量申请一片初始化为零的空间 | | | | | |
| 五、实验总结  请回答：   1. 实验过程中所遇到的问题及解决办法   实验一：  （1）刚开始的时候运行失败，提示找不到makefile，问题来源于我将mingw/bin目录下的可执行文件改成了make.exe，后面将名字改成mingw32-make.exe后才能编译成功。所以使用指令时只能使用mingw32-make来代替make指令。并且需要把test文件里build.py中的make也改成mingw32-make才能编译测试程序。  （2）名字中带’\_’字符的出现问题，解决方案是在empty状态的对应判断条件中加入’\_’。  语法分析：  函数确少返回值以及返回false；  （2）log输出只有一个in parse；  原因：开始的时候root初始化方式出错。  frontend::CompUnit\* Parser::get\_abstract\_syntax\_tree(){  auto\* root = new CompUnit();  parseCompUnit(root);  return root;  }  （3）UnaryExp -> PrimaryExp | Ident '(' [FuncRParams] ')' | UnaryOp UnaryExp文法进入PrimaryExp失败原因是判断条件没有写完整，解决方案是将这个分支放到else中。  （4）return a+b;这个语句中在parseUnaryExp时没有正确的进入parseprimary中，原因是有两个地方开头都ident没有识别出来。  解决方法：优化逻辑。  （5）第21个测试点陷入死循环：原因是parsestmt函数中的一个分支逻辑错误，将 /\* Stmt -> Lval '=' Exp \*/分支逻辑修正。  （6） b = func(a);语句在parseunaryexp中错误的进入了primaryexp中  解决：在**dent** '(' [FuncRParams] ')' 时没有把dent加入token导致后续判断失败  （7）while (a < 20) 错误  RelExp -> AddExp { ('<' | '>' | '<=' | '>=') AddExp }文法没有把符号添加  （8）单个;死循环：发现时stmt中的[Exp] ';' 出现问题  解决：把exp情况分开判断。  实验二：  1.开始的时候重新测试测s0和s1发现出现问题，原因是有一个浮点数是.1token中没有判断这个逻辑，修正即可。  第二个问题是读取sy时的注释处理问题，修复后重新测试，s0和s1都成功。  2.在这个函数中，获取父节点失败。  void frontend::Analyzer::analysisVarDef(frontend::VarDef \*root, vector<ir::Instruction \*> &buffer) {  std::printf("enter analysisVarDef\n");  ir::Type type= dynamic\_cast<ConstDecl\*>(root->parent)->t;  std::printf("i get the parent");  解决：  语法分析时没有把parent赋值，所以现在只能把函数的参数修改，增加一个参数，在上一级把参数传递。  void frontend::Analyzer::analysisVarDef(frontend::VarDef \*root, vector<ir::Instruction \*> &buffer,ir::Type type) {  3  varSte.operand=ir::Operand(symbol\_table.get\_scoped\_name(varName),type);  这个代码会导致table插入出错  解决：先把原名插入再改名；  4这一段代码只能分析一个compUnit没有进入第二个  int a = 3;  int b = 5;  解决：排查问题后发现， std::cout<<"varName:"<<varName<<"scname:"<<symbol\_table.get\_scoped\_name(varName)<<std::endl;  重命名的名字为空，导致Instruction创建失败，程序卡住   1. 对实验的建议   希望能增加一些实验教学。 | | | | | |
|  | | | | | |
|  | | | | | |