《编译技术》课程设计文档

## 一．需求说明

### 1．文法说明

获取的文法为扩充的C0文法，具体请看如下网页

<http://www.menghuanlater.cn/C0.jsp>

文法解读参照文法解读阶段的文档

对于文法的相关改写(削除左递归,可用递归下降子程序法实现语法分析):

暂无(由于给定的文法初版就已经大量的使用扩充的文法表示形式，消除了

许多的左递归情况，目前暂未发现存在左递归的文法)

### 2．目标代码说明

目标代码生成为基于MIPS指令集的32位汇编代码。

具体采用的指令集下载地址:

<https://github.com/menghuanlater/BUAA_Complier_Design/blob/master/MIPS32.pdf>

### 3. 优化方案

优化方案:通过DAG图法删除基本块内公共子表达式；通过数据流分析建立

活跃冲突图，使用着色算法进行全局寄存器的分配;循环结构优化(暂没有详

细的算法)

## 二．详细设计

### 1．程序结构

图片包含 文字

已生成高可信度的说明

### 2．类/方法/函数功能(描述各类/方法或函数的功能，以及关键算法)

全局常量的相关定义:

#define KEY\_NUM 14 //保留字数量定义

#define SYM\_NUM 38 //记忆符数量

//枚举记忆符对应的类别编码

enum SymbolCode{

CONSTSY, INTSY, CHARSY, VOIDSY,

MAINSY, IFSY, ELSESY, SWITCHSY,

CASESY, DEFAULTSY, WHILESY, SCANFSY,

PRINTFSY, RETURNSY, IDENTIFIER, INTNUM,

STRING, CHAR, ADD, SUB,

MULT, DIV, LESS, LESSEQ,

EQUAL, MOREEQ, NOTEQ, MORE,

COMMA, COLON, SEMI, LSBRACKET,

RSBRACKET, LMBRACKET, RMBRACKET, LBBRACKET,

RBBRACKET, ASSIGN

};

//保留字

const char \* keyWordsArr[KEY\_NUM] = {

"const", "int", "char", "void",

"main" , "if" , "else", "switch",

"case" , "default", "while", "scanf",

"printf", "return"

};

//记忆符(凡是带SY后缀的都是保留字的记忆符)

const char \* SymbolArr[SYM\_NUM] = {

"CONSTSY", "INTSY", "CHARSY", "VOIDSY",

"MAINSY", "IFSY", "ELSESY", "SWITCHSY",

"CASESY", "DEFAULTSY", "WHILESY", "SCANFSY",

"PRINTFSY", "RETURNSY", "IDENTIFIER", "INTNUM",

"STRING", "CHAR", "ADD", "SUB",

"MULT", "DIV", "LESS", "LESSEQ",

"EQUAL", "MOREEQ", "NOTEQ", "MORE",

"COMMA", "COLON", "SEMI", "LSBRACKET",

"RSBRACKET", "LMBRACKET", "RMBRACKET", "LBBRACKET",

"RBBRACKET", "ASSIGN"

};

-----------------------------------------------------------------------

词法分析：

词法分析为LexicalAnalysis类

类属性成员如下：

string fileContents;//编译文件的全部内容

    long fileLength;//编译文件的大小

int index; //索引

char globalChar;//存放当前读进的字符

string globalString;//存放当前读进的字符串

int globalNumber;//存放当前读进的整数

enum SymbolCode globalSymbol;//当前所识别单词的类型

static const int maxWordLength = 1024;//一个单词最大长度

int lineCount;//行计数器,目的在于发现非法字符提示所在文件的行数

const Error & myError;//错误处理类引用,全局公用

类的各函数（方法）

(1)retract:回退一个字符

(2)getChar: 读取下一个字符

(3)toLow:由于文法说明规定了标识符不区分大小写,所以此函数的目的是为了统一化处理,将标识符一律小写化

(4)reserver:查找当前识别出的标识符是否是保留字,如果是，返回对应保留字

在全局保留字数组中的下标+1,否则返回0.

(5) LexicalAnalysis(const Error & error);构造函数

(6)readFile: 读取文件的全部内容到fileContents中

(7)nextSym:语法分析程序调用,读取下一个单词

(8) getGlobalChar:返回读取到的单字符

(9)getGlobalNum:返回读取到的数字

(10)getGlobalString:返回读取到的字符串或者标识符

(11)getGlobalSymbol:返回读取到的单词的记忆符编码

错误处理：(暂时包括词法分析):

错误处理类为Error

类属性：

类成员函数(方法):

1. LexicalAnalysisError:词法分析过程发现的错误信息处理
2. SyntaxAnalysisError:语法分析过程中发现的错误信息处理

### 3．调用依赖关系(说明各类之间的关系，方法/函数之间的调用关系)

### 4．符号表管理方案

【说明符号表的数据结构、管理算法】

### 5．存储分配方案

【说明运行时的存储组织及管理方案，运行栈结构】

### 6. 四元式设计\*

【对采用的四元式进行详细说明】

### 7. 目标代码生成方案\*

【说明代码生成有关的数据结构、关键算法】

### 8. 优化方案\*

【说明代码优化有关的数据结构、关键算法】

### 9. 出错处理

1.词法分析的出错处理:词法分析出错点有以下几种情况：单字符的值不在文法范围内,单字符不是类似于’a’的结构,字符串中出现定义范围以外的字符，字符串的左右双引号不匹配，源文件中出现非法字符(一般为中文字符、日文字符等等).对于前几种错误,会直接报错，并跳过相关局部分析其他部分，对于文本中出现中文日文阿拉伯文等情况直接停止分析,提示用户修改掉相关的字符

## 三．操作说明

### 1．运行环境

工程项目所属：VS2010

中间代码运行所需：Mars4\_4及更高版本

运行平台：windows操作系统，JDK1.7或者更高版本

### 2．操作步骤

1.使用VS2010打开项目工程，编译工程，运行程序

2.在控制台输入编译文件的绝对路径，进行编译

3.源文件正确编译，生成mips32指令集支持的汇编中间代码，如果源文件

存在bug，将不生成中间代码文件，代替为在控制台输出错误信息

4.通过Mars4\_4或更高版本仿真器进行中间代码的模拟运行。

## 四．测试报告

(整体编译器全部设计最终实现进行全面覆盖测试检查时补充此处)

### 1．测试程序及测试结果

【给出提供的测试程序以及每个程序的测试结果，至少5个正确程序，5个错误程序，无需截屏】

### 2．测试结果分析

【说明上述测试程序对语法成分的覆盖情况】

## 五．总结感想

整体编译器全部设计最终实现时补充感想。