**BC/NW 2016 № 2 (29):11.3**

**РАЗРАБОТКА ЛЕКСИЧЕСКОГО АНАЛИЗАТОРА КОДА SYSTEMC**

Цепаев В. С.

Цель работы **–** разработать программу, производящую посимвольное считывание символов из модели, выполненной на языке SystemC, распознающую среди них отдельные слова (операции, операнды и служебные) и определяющую, к какой категории они относятся.

Исходными данными являются пары файлов: заголовочный и текста модели, представляющие в совокупности модель цифрового устройства, выполненную на языке SystemC.

Основой алгоритма лексического анализа является прочтение исходного текста построчно, определение в строке слов языка SystemC и определение категории в которую эти слова входят.

Для реализации алгоритма в прикладной программе был использован язык программирования С#.

**Описание структуры модели языка SYSTEMC**

***Описание файловой структуры языка SystemC***

Проект на языке SystemC состоит из набора файлов .cpp и .h. Для того чтобы полноценно работать, не требуется какая-либо среда разработки,поддерживающая данный язык. Достаточным условием является наличиекомпилятора C++ и библиотеки SystemC. На рисунке 1.1 изображена схемапроцесса компиляции проекта на SystemC.



Рисунок 1.1 – Процесс компиляции проекта SystemC

Компилятор C++ обрабатывает файлы .h и .cpp из набора файлов проекта SystemC и создает объектные файлы (расширение файла .o). После создания объектных файлов компоновщик связывает созданные объектные файлы и соответствующие объектные файлы библиотеки SystemC (и других библиотек, таких как стандартная библиотека шаблонов или STL). В результате создается исполняемый файл, который содержит в себе ядро моделирования SystemC с функциональными возможностями исходного проекта.

В данной разработке рассматриваться только содержимое файлов .h и .cpp .

***Описание модельной структуры.***

Модель цифрового устройства на языке SystemC состоит из нескольких блоков, у каждого из которых, в свою очередь, есть своя структура. Для описания каждой составной части модели используется макрос С++ SC\_MODULE.

Описание каждого модуля на SystemC содержится в заголовочном файле <имя модуля>.h и содержит:

* Объявление портов;
* Объявление внутренних сигналов;
* Объявление внутренних переменных;
* Объявление экземпляров подмодулей;
* Конструктор;
* Деструктор;
* Процессы;
* Функции процессов;
* Вспомогательные функции.

Поскольку в данном курсовом проекте рассматривается только лексическая

часть анализатора кода, то весь будет разбит на следующие лексические

категории:

* Служебные слова языка SystemC и языка С++;
* Типы данных;
* Структурных единиц кода (переменные, константы и функции);
* Комментарии

**Разработка программы**

***Пользовательский интерфейс***

Программы написана с помощью языка C# и представляет собой исполняемый файл Windows Forms, в котором реализованы диалог открытия файлов .cpp или .h , лексический анализ открытого файла с выводом результата на экран, а так же возможность сохранить результат анализа в файле .txt посредством диалога сохранения файла. Каждая из перечисленных функций реализуется путем нажатия на соответствующую кнопку.



Рис. 2. Внешний вид программы.

Таблица 1

Элементы управления программы.



Обработка нажатий на кнопки button1, button3 и button4 не будет рассматриваться в данной разработке по причине их тривиальности.

По нажатию кнопки «Анализировать» проверяется существование файла, указаного в верхней текстовой форме. Путь к файлу загружается из диалога выбора файла, но пользователь может изменить этот путь вручную.

В случае, если файл не существует, выдается сообщение об этом. В случае если файл существует, начинается процедура анализа.

***Анализ файла***



Рис. 2.2 Схема алгоритма анализа файла

В верхнем уровне алгоритма анализа файла происходит чтение файла по строкам и вызов процедуры анализа строки для каждой из считанных строк.

По достижению конца файла в результат анализа добавляется запись о завершении анализа файла с полным путем к файлу, и активизируется кнопка сохранения результата.

2.3***Анализ строки***



Рис. 2.3. Схема алгоритма анализа строки

Анализируемая строка считывается посимвольно и накапливает символы в слове до тех пор, пока не произойдет одна из ситуаций:

1) Встретится символ окончания слова. В эти символы входят пунктуационные символы, которые не могут входить в слово языка SystemC. По нахождению такого символа накопленное слово отправляется на анализ.

2) Cтрока завершится. Последний символ строки добавляется к слову, если он не является символом окончания слова, и слово отправляется на анализ.

Отдельно обрабатываются комбинации символов окончания слова:

1)Комбинация «//» означает открытие комментария. По нахождению такой комбинации поднимает флаг «комментарий», с которым любой дальнейший символ считается частью слова.

2)Комбинации «>>» и «<<» означают операцию перегрузки. По их появлению в результат анализа добавляется соответствующая запись.

***Анализ слова***



Рис 2.4 Схема алгоритма анализа слова

Анализируемое слово проходит проверку на соответствие одной из категорий по следущим признакам:

Таблица 2.

Категории слов



**Тестирование программы**

В качестве проверки работоспособности был использован заголовочный файл модели адресного регистра ADRRG.h

Содержимое файла ADRRG.h :

#ifndef \_\_ADRRG\_h\_\_

#define \_\_ADRRG\_h\_\_

#include <systemc.h>

SC\_MODULE( ADRRG )

{

sc\_in< sc\_bv< 14 > > A;

sc\_in <bool> C1;

sc\_out< sc\_bv< 10 > > MA;

sc\_out< sc\_bv< 4 > > CA;

sc\_event delay;

SC\_CTOR( ADRRG ):

A("A"),

C1("C1"),

MA("MA"),

CA("CA")

{

SC\_METHOD(do\_delay);

sensitive << C1;

SC\_METHOD(do\_ADRRG);

sensitive << delay;

}

void do\_delay(){delay.notify(25, SC\_NS);}

void do\_ADRRG()

{

MA.write(A.read().range(9,0));

CA.write(A.read().range(13,10));

}

~ADRRG()

{

}

};

SC\_MODULE\_EXPORT( ADRRG )

#endif //\_\_ADRRG\_h\_\_

Содержимое файла результата анализа ADDRG\_result.txt :

Начало анализа файла C:\Users\lenovo\Desktop\МЭИ\Masters Thesis\master\_exec\ADRRG.h

Анализ новой строки: 1

#ifndef служебное слово

\_\_ADRRG\_h\_\_ операнд или метод

Анализ новой строки: 2

#define служебное слово

\_\_ADRRG\_h\_\_ операнд или метод

Анализ новой строки: 3

Анализ новой строки: 4

#include служебное слово

systemc операнд или метод

Анализ новой строки: 3

Анализ новой строки: 4

#include служебное слово

systemc операнд или метод

h операнд или метод

Анализ новой строки: 5

Анализ новой строки: 6

SC\_MODULE служебное слово

ADRRG операнд или метод

Анализ новой строки: 7

Анализ новой строки: 8

Анализ новой строки: 9

sc\_in служебное слово

sc\_bv тип данных

14 операнд или метод

A операнд или метод

Анализ новой строки: 10

sc\_in служебное слово

bool тип данных

C1 операнд или метод

Анализ новой строки: 11

sc\_out служебное слово

sc\_bv тип данных

10 операнд или метод

MA операнд или метод

Анализ новой строки: 12

sc\_out служебное слово

sc\_bv тип данных

4 операнд или метод

CA операнд или метод

Анализ новой строки: 13

sc\_event служебное слово

delay операнд или метод

Анализ новой строки: 14

Анализ новой строки: 15

SC\_CTOR служебное слово

ADRRG операнд или метод

Анализ новой строки: 16

A операнд или метод

"A" операнд или метод

Анализ новой строки: 17

C1 операнд или метод

"C1" операнд или метод

Анализ новой строки: 18

MA операнд или метод

"MA"

Анализ новой строки: 19

CA операнд или метод

"CA" операнд или метод

Анализ новой строки: 20

Анализ новой строки: 21

SC\_METHOD служебное слово

do\_delay операнд или метод

Анализ новой строки: 22

sensitive служебное слово

C1 операнд или метод

Анализ новой строки: 23

SC\_METHOD служебное слово

do\_ADRRG операнд или метод

Анализ новой строки: 24

sensitive служебное слово

delay операнд или метод

Анализ новой строки: 25

Анализ новой строки: 26

void служебное слово

//\_\_ADRRG\_h\_\_ комментарий

Анализ новой строки: 42

Достигнут конец файла C:\Users\lenovo\Desktop\МЭИ\Masters Thesis\master\_exec\ADRRG.h

**Заключение**

В результате выполнения разработки был получен алгоритм лексического анализа кода на языке SystemC. Данный алгоритм был реализован в виде прикладной программы для компьютеров под управлением операционной системы Microsoft Windows. Работоспособность алгоритма была протестирована с использованием уже существующих моделей цифровых устройств.

Данный алгоритм является базой для дальнейшей разработки анализатора кода SystemC , который будет являться частью транслятора с языка SystemC на язык VHDL.