

УДК 519.682.1: 681.142.2

А.П. СОБЧАК, К.В. ХОДАРЕВ

*Национальный аэрокосмический университет им. Н.Е. Жуковского "ХАИ", Украина***МОДЕЛЬ АППАРАТНОЙ РЕАЛИЗАЦИИ АЛГОРИТМА РУТИСХАУЗЕРА**

В статье рассматриваются алгоритмы трансляции выражений на примере алгоритма Рутисхаузера. Приводится один из способов его аппаратной реализации.

алгоритм, трансляция выражений, аппаратная реализация, модель

Введение

Алгоритм Рутисхаузера является одним из алгоритмов трансляции выражений. Данный алгоритм является одним из самых ранних. Его особенностью является предположение о полной скобочной структуре выражения.

Под полной скобочной записью выражения понимается запись, в которой порядок действий задается расстановкой скобок. Неявное старшинство операций при этом не учитывается [1].

Формулирование проблемы. Обработывая выражение с полной скобочной структурой, алгоритм присваивает каждому символу из строки номер уровня по следующему правилу:

- 1) если это открывающаяся скобка или переменная, то значение увеличивается на 1;
- 2) если знак операции или закрывающая скобка, то уменьшается на 1.

Для выражения $(A+(B+C))$ присваивание значений уровня будет происходить так, как это представлено в табл. 1.

Таблица 1
Присваивание значений уровня

№ символа	1	2	3	4	5	6	7	8	9
Символы строки	(A	+	(B	*	C))
Номера уровней	1	2	1	2	3	2	3	2	1

Алгоритм складывается из следующих шагов:

- 1) выполнить расстановку уровней;
- 2) выполнить поиск элементов строки с максимальным значением уровня;
- 3) выделить тройку – два операнда с максимальным значением уровня и операцию, которая заключена между ними;
- 4) результат вычисления тройки обозначить вспомогательной переменной;
- 5) из исходной строки удалить выделенную тройку вместе с ее скобками, а на ее место поместить вспомогательную переменную, обозначающую результат, со значением уровня на единицу меньше, чем у выделенной тройки;
- 6) выполнять пункты 2 – 5 до тех пор, пока во входной строке не останется одна переменная, обозначающая общий результат выражения [2].

**Решение проблемы.
Аппаратная реализация
алгоритма Рутисхаузера**

На основе алгоритма Рутисхаузера реализована аппаратная модель транслятора выражений с использованием языка аппаратного описания VHDL [3 – 6].

Сначала нужно сформировать строку уровней. Алгоритм формирования строки уровней представлен на рис. 1.

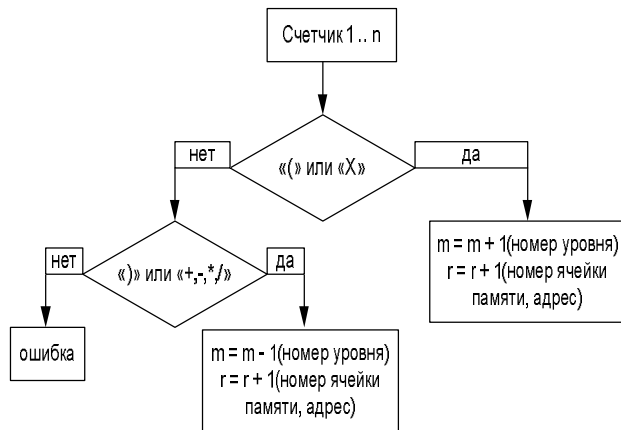


Рис. 1. Алгоритм формирования строки уровней

Дальнейшие действия представлены в виде алгоритма на рис. 2.

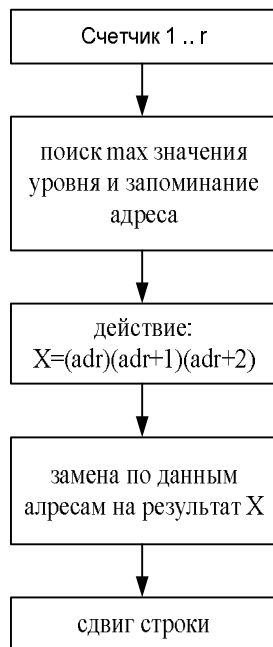


Рис. 2. Продолжение алгоритма формирования строки уровней

Далее повторяем оба алгоритма до тех пор, пока в строке не останется одно значение – результат действий.

Опишем структуру транслятора, реализованного на языке VHDL более подробно.

В качестве хранителя входной информации (выражения) используется блок памяти из библиотеки стандартных элементов программы MAX + plus II. Все элементы перебираются при помощи счетчика

count и подаются на выход памяти, где анализируются блоком *select*. В зависимости от того, какой это элемент, принимается решение о том, увеличивать или уменьшать уровень на единицу. За это отвечает блок *blockdown*.

Для поиска максимального элемента используется блок *max_elem*, который в результате своей работы выдает адрес элемента, имеющего максимальный уровень, обозначим его *addrMax*.

Следующий этап заключается в том, чтобы взять элементы с адресами *addrMax*, *addrMax + 1* и *addrMax + 2* и произвести над ними арифметическое действие, код которого записан по адресу *addrMax + 1*. Эти операции производятся в блоке *deystvie1*. Итогом его работы является результат арифметического действия.

Затем необходимо осуществить замену. Она производится следующим образом: результат арифметического действия записывается по адресу *addrMax - 1*, а по адресам от *addrMax* до *addrMax + 3* записываются нули, которые при повторном анализе строки программой не воспринимаются. Данные действия производятся в блоках *zamena* и *zamena1*.

Описание работы каждого блока представлено ниже, язык реализации – VHDL.

Блок *Count* включает в себя два счетчика: первый считает от 0 и далее, т.е. перебирает значения из памяти, второй считает исходя из полученного позднее адреса элемента, имеющего максимальный уровень. Также существует возможность сброса счетчика по приходу сигнала *reset*.

Временная диаграмма работы блока *count* представлена на рис. 3.

Блок *select* производит следующие действия: если читается открывающаяся скобка или число, на выходе *cnt* – единица, если закрывающаяся скобка или арифметическое действие, то на выходе *cnt1* тоже единица.

Временная диаграмма работы блока *select* представлена на рис. 4.

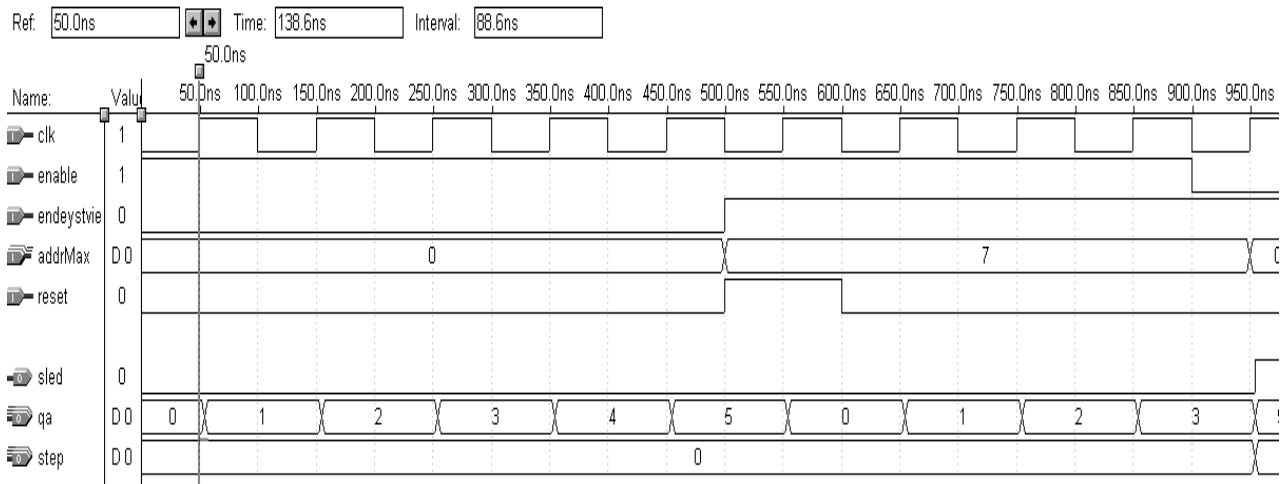


Рис. 3. Временная диаграмма работы блока *count*

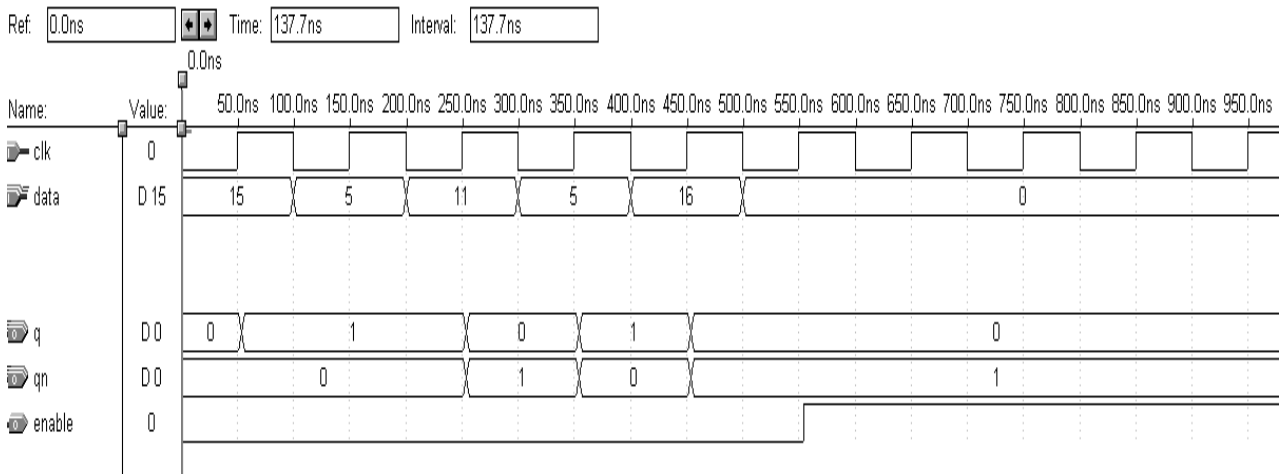


Рис. 4. Временная диаграмма работы блока *select*

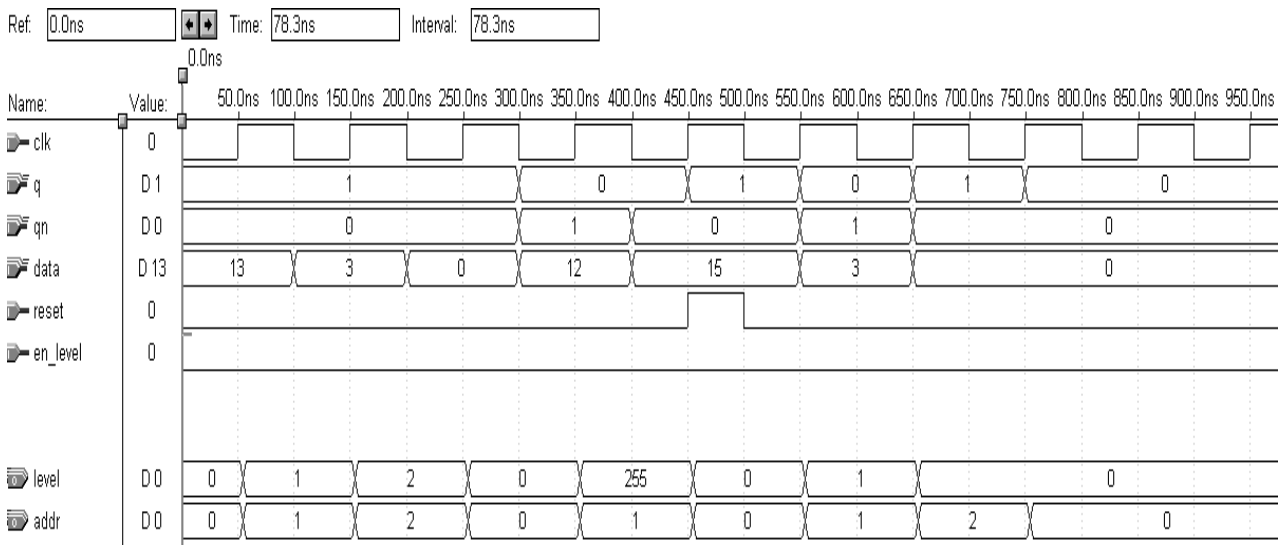


Рис. 5. Временная диаграмма работы блока *blockdown*

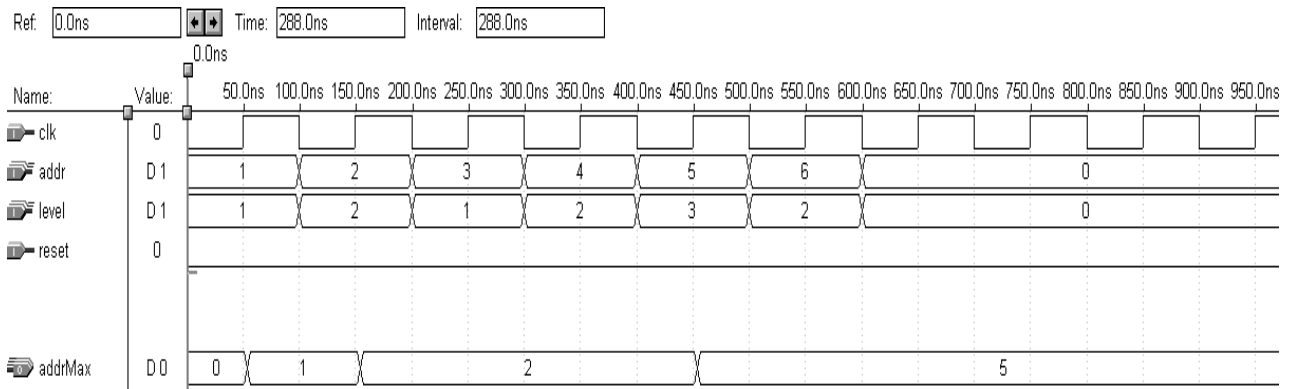


Рис. 6. Временная диаграмма работы блока *max_elem*

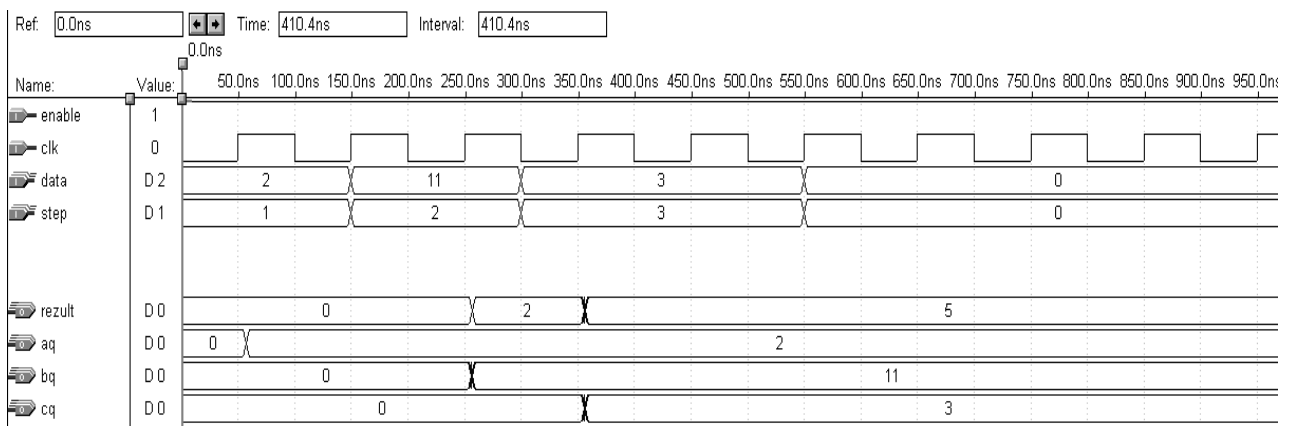


Рис. 7. Временная диаграмма работы блока *deystvie1*

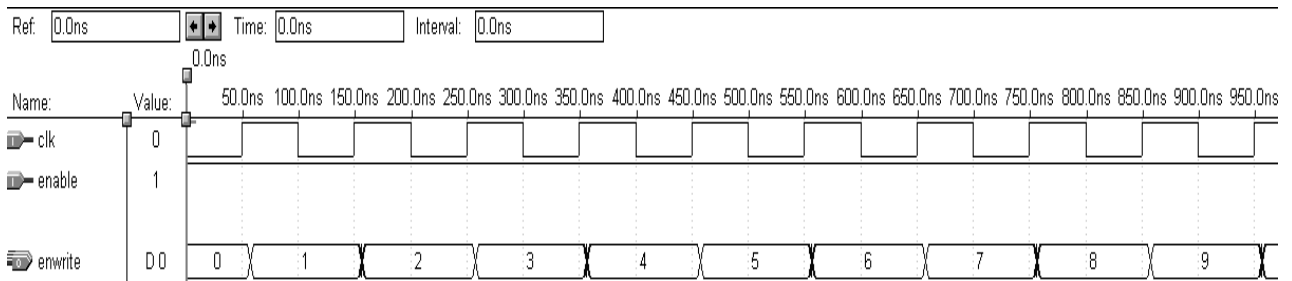


Рис. 8. Временная диаграмма работы блока *zamena*

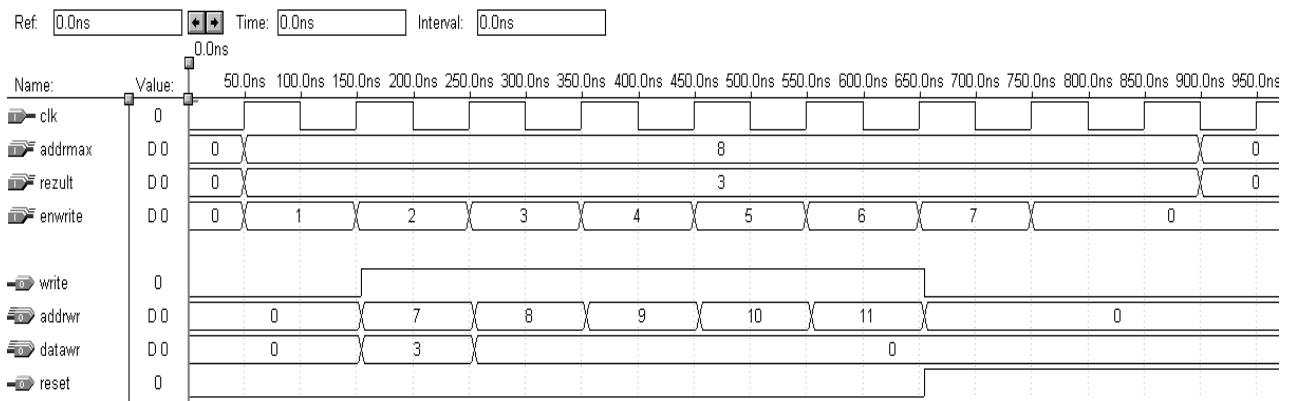


Рис. 9. Временная диаграмма работы блока *zamena1*

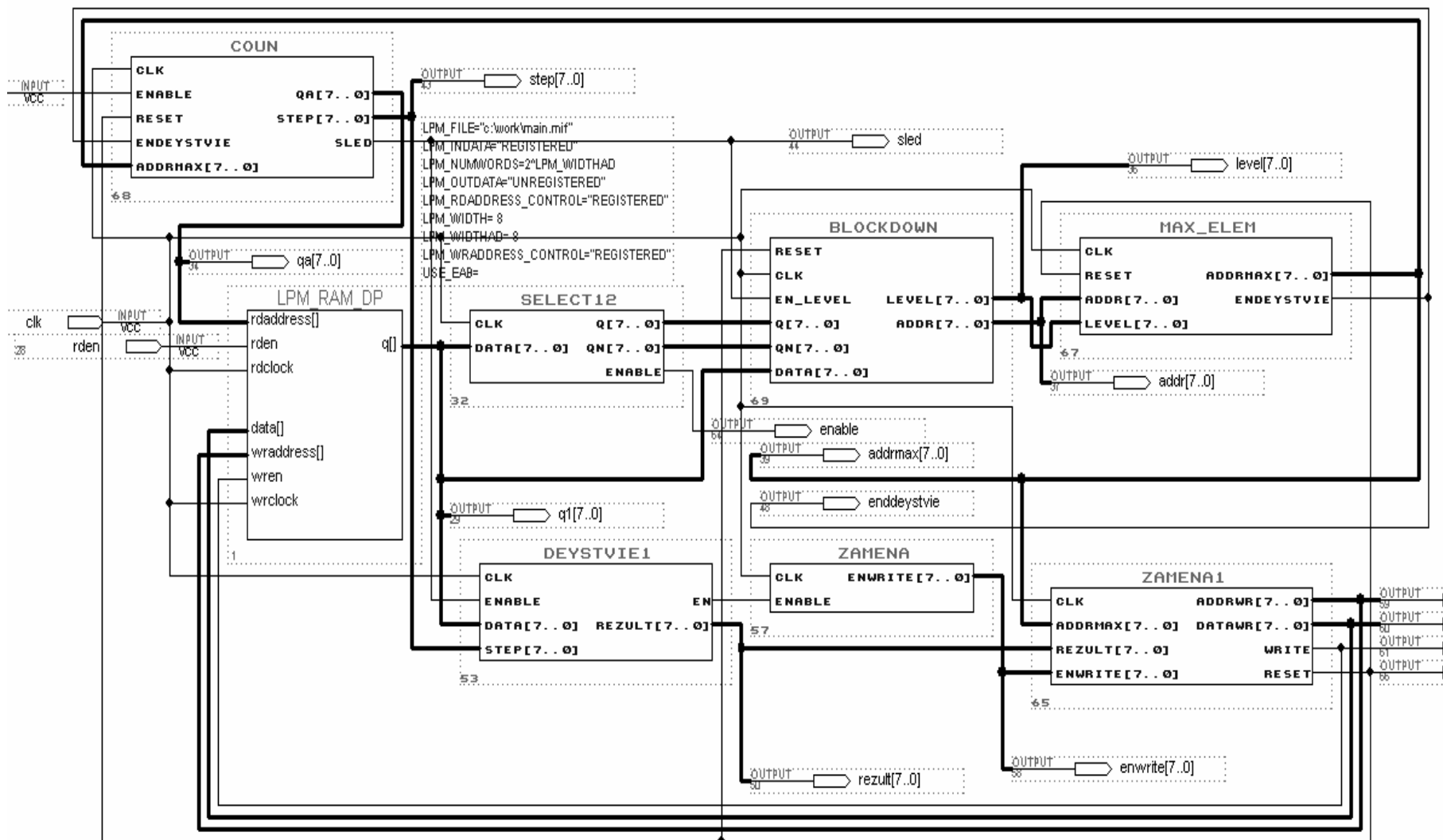


Рис. 10. Вид устройства в графическом редакторе системы MAX+PLUS II

Блок *blockdown* непосредственно формирует строку уровней, основываясь на данных, поступающих из блока *select*. Временная диаграмма работы блока *blockdown* представлена на рис. 5.

Блок *max_elem* анализирует строку уровней и находит адрес элемента с максимальным уровнем. Временная диаграмма работы блока *max_elem* представлена на рис. 6.

Блок *deystvie1* производит арифметическое действие с элементами, находящимися по адресам *max_elem*, *max_elem + 1* и *max_elem + 2* и передает результат для дальнейшей обработки блоку *zamenal*, который записывает результат по адресу *max_elem - 1*, а по адресам, начиная от *max_elem* и до *max_elem + 3*, записываются нули, которые при дальнейшей обработке строки не воспринимаются. Блок *zamenal* представляет собой счетчик, координирующий действия блока *zamenal*. Временные диаграммы работы блоков *deystvie1*, *zamenal* и *zamenal* представлены на рис. 7 – 9 соответственно.

Сигнал *reset* предназначен для сброса всей системы в начальное состояние и используется при каждой повторной обработке строки вплоть до достижения конечного результата.

Загрузка данных, т.е. изначальной скобочной структуры, для удобства производится из текстового файла непосредственно в память, но возможна и последовательная загрузка данных перед началом работы устройства.

Вид всего устройства в графическом редакторе системы MAX + PLUS II представлен на рис. 10.

Заключение

Таким образом, разработана модель алгоритма трансляции выражений на базе алгоритма Рутисхаузера. Полученное устройство, преобразуя входные данные, выдает соответствующий результат, т.е. в данном случае результат вычисления. Реализован-

ный алгоритм является одним из простейших алгоритмов трансляции данных и поэтому существует необходимость в поиске, создании более сложных и более эффективных алгоритмов для трансляции более сложных входных данных, например, программного кода и построении их аппаратных моделей, и, в конечном счете, увеличении скорости работы по сравнению с их программными аналогами [7, 8].

Литература

1. Брежнев А.М. Конспект лекций по дисциплине «Системное программное обеспечение», тема: «Трансляторы». – Северодонецк, 1995. – 45 с.
2. Романов Е.Л. Основы построения трансляторов (конспект лекций). – Н.: НГУ, 1996. – 103 с.
3. Стешенко В.Б. ПЛИС фирмы ALTERA: проектирование устройств обработки сигналов. – М.: ДОДЕКА, 2000. – 128 с.
4. Жихарев В.Я., Илюшко В.М., Чумаченко И.В. Проектирование электронных компиляторов: Монография. – Х.: Факт, 1999. – 88 с.
5. Антонов А.П. Язык описания цифровых устройств AlteraHDL: Практический курс. – М.: ИП «Радиософт», 2001. – 224 с.
6. Соловьев В.В. Проектирование цифровых систем на основе ПЛИС // Горячая линия – Телеком, 2001. – 265 с.
7. Шальто А.А. Методы аппаратной и программной реализации алгоритмов – С-Пб.: Наука, 2000. – 749 с.
8. Собчак А.П., Марченко А.Н., Ходарев К.В. Реализация лексического анализатора на языке аппаратного описания. – Х.: Нац. аэрокосм. ун-т «ХАИ», 2004. – С. 116 – 121.

Поступила в редакцию 31.10.2005

Рецензент: д-р техн. наук, проф. В.М. Илюшко, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков.