УДК 519.714.5
doi: 10.18101/2304-5728-2017-3-64-72

РЕАЛИЗАЦИЯ БУЛЕВЫХ ФУНКЦИЙ КОМБИНАЦИОННЫМИ СХЕМАМИ С ПОМОЩЬЮ ФУНКЦИОНАЛЬНЫХ ГРАММАТИК

© Чимитов Василий Валерьевич
аспирант,
Восточно-Сибирский государственный университет технологий и управления
Россия, 670013, г. Улан-Удэ, ул. Ключевская, 40В
E-mail: cvh1304@yandex.ru

© Могнонов Петр Борисович
кандидат технических наук, доцент,
Восточно-Сибирский государственный университет технологий и управления
Россия, 670013, г. Улан-Удэ, ул. Ключевская, 40В
E-mail: pmognonov@mail.ru

В статье предлагается метод декомпозиции булевых функций с помощью функциональных грамматик. Рассматриваются основные вопросы построения функциональных грамматик для реализации булевых функций комбинационными схемами. Демонстрируется порождающая способность функциональных грамматик на примере синтеза комбинационной схемы.

Ключевые слова: грамматика; комбинационные схемы; декомпозиция булевых функций.

Введение

При современных темпах развития полупроводниковых технологий существенно усложняются функции, возлагаемые на средства их автоматического проектирования. Сложность и высокая стоимость разработки цифровых систем стимулируют развитие новых более универсальных методов и средств автоматизации проектирования цифровых устройств.

Теоретические основы и общеизвестные классические методы логического проектирования представлены в работах [1; 2]. В работе [3] дан обзор, где задачу логического проектирования сводят к задаче декомпозиции булевых функций. В работе [4] рассматриваются методы логического синтеза структур программируемых логических интегральных схем, а в [5] предложена методика синтеза комбинационных схем с использованием искусственных сетей. В этом подходе решение задачи логического проектирования сводится к обучению исходно избыточной бинарной многослойной нейронной сети. При этом наиболее «естественной» средой для реализации такого подхода является нейрокомпьютер, построенный из цифровых компонентов, обеспечивающих возможность настройки его на конкретные задачи, представленные в нейросетевом базисе.

В данной работе предлагается метод синтеза комбинационной схемы, реализующей заданную булеву функцию с помощью аппарата функциональных грамматик. Этот аппарат в дальнейшем позволит автоматизировать процесс генерации программ на языке описания аппаратуры на основе функционального представления цифрового устройства.

1. Функциональная грамматика, определение и свойства

В работах [6; 7] вводится тезис о том, что любая предметно-ориентированная теория является полиморфной программой, написанной на специализированном естественном языке. Согласно данному подходу с использованием функциональных грамматик можно рассматривать любую теорию как алгебраическую систему $(M, f_1, f_2, ..., f_n)$, где M — множество, в разных теориях имеющее различное название: базис элементов, структура данных, семантическая структура, глубинная структура и т. п. При этом всякое предложение естественного языка является закодированной суперпозицией (или множеством суперпозиций) функций f_i.

В качестве функциональной грамматики понимается множество формальных грамматик следующего вида:

$$G = (V_T, V_N, P, F, S),$$

где V_T — конечное множество терминальных символов, V_N — конечное множество нетерминальных символов, F — конечное множество функций вывода; P – конечное множество правил вывода, множество упорядоченных пар следующего вида: $A \to \alpha\{f\}, A \in V_T \cup V_N$, α — цепочка символов из $V_T \cup V_N$, f — функция $f \in F$, S — начальный символ $S \in V_N$.

Для решения задачи синтеза цифрового устройства приведем функциональную грамматику G_0, используя некоторые разделы теории проектирования цифровых систем и дискретной математики. В качестве входных данных функциональной грамматики рассматривается система булевых функций в дизъюнктивной нормальной форме, представленных операциями: $\neg$ (отрицание), $\cap$ (конъюнкция), $\cup$ (дизъюнкция). Выводу функциональной грамматики будет соответствовать суперпозиция функций. Существует несколько вариантов использования полученного результата:

1. Вывести суперпозицию функций на понятном пользователю языке, либо на одном из языков описания аппаратуры (HDL — hardware description language).

2. Вычислить суперпозицию функций с заданными исходными параметрами для моделирования работы цифрового устройства.

Построение функциональной грамматики начинается с составления алфавита $V = V_T \cup V_N$. Для этого выделим основные понятия для системы булевых функций, сопоставив им символы алфавита (табл. 1).

Таблица 1

Алфавит символов грамматики

№	Понятие теории проектирования цифровых систем и дискретной математики	Символ алфавита
1	Исходное задание	<Задание>
2	Базис операций	<Базис>
3	Система булевых функций $\{f_1, f_2, ..., f_m\}$	<СБФ>
4	Булева функция $f_m, m = \overline{0, N}$	<Функция>
5	Выходной аргумент функции f	<Выход>
6	Дизъюнкция, логическая сумма логических произведений аргументов функции	<Дизъюнкция>
7	Конъюнкция, логическое произведение аргументов функции	<Конъюнкция>
8	Аргумент функции	<Аргумент>
9	Буквенное обозначение аргумента функции	<Символ>

Алфавит символов примет следующий вид:

$$V = \left\{ \begin{array}{c} < Задание >, < СБФ >, < Базис >, < Функция >, < Выход >, \\ < Дизъюнкция >, < Конъюнкция >, < Аргумент >, < Символ > \end{array} \right\}.$$

Для символов из алфавита составим правила вывода:

$$< Задание > \to < СБФ >; < Базис > \{f_0\}$$

$$< СБФ > \to < Фунция >, < СБФ >; \{f_1\} \mid < Функция >; \{f_2\}$$

$$< Функция > \to < Выход > = < Дизъюнкция > \{f_3\}$$

$$< Дизъюнкция > \to \neg < Дизъюнкция > \{f_4\} \mid$$

$$< Конъюнкция > + < Дизъюнкция > \{f_5\} \mid < Конъюнкция > \{f_2\}$$

$$< Конъюнкция > \to \neg < Конъюнкция > \{f_4\} \mid$$

$$< Аргумент > \cdot < Конъюнкция > \{f_6\} \mid < Аргумент > \{f_2\}$$

$$< Аргумент > \to \neg < Символ > \{f_4\} \mid < Символ > \{f_2\}$$

$$< Символ > \to a \mid b \mid c \mid d \mid ... \mid z \{f_7\}$$

$$< Выход > \to < Символ > \{f_2\}$$

$$< Базис > \to И, НЕ \mid ИЛИ, НЕ \mid И, ИЛИ, НЕ \mid И - НЕ \mid ИЛИ - НЕ \{f_8\}$$

Функции f_i:

$$f_0 = (\underline{функ}\, x, \underline{знач}\, y)\underline{знач} : (X)\, eval\, X$$

$$f_1 = (\underline{функ}\, x, y)\underline{функ} : (X, Y) X, Y$$

$$f_2 = (\underline{функ}\, x)\underline{функ} : (X)\, X$$

$$f_3 = (\underline{функ}\, x, y)\underline{функ} : (X, Y)\, X(Y)$$

$$f_4 = (\underline{конт}\, t, \underline{функ}\, x)\,\underline{функ} : (T:4, X)\, f_4^0(X,X) \,\big|\, (T:3, X)\, f_5^0(X,X) \,\big|$$
$$(T:0\,|\,1\,|\,2, X)\, f_1^0(X)$$
$$f_5 = (\underline{конт}\, t, \underline{функ}\, x, y)\,\underline{функ} : (T:4, X, Y)\, f_4^0(f_4^0(X,Y), f_4^0(X,Y)) \,\big|$$
$$(T:3, X, Y)\, f_5^0(f_5^0(X,X), f_5^0(X,X)) \,\big|\, (T:0, X, Y)\, f_1^0(f_3^0(f_1^0(X), f_1^0(Y))) \,\big|$$
$$(T:1\,|\,2, X, Y) f_2^0(X,Y)$$
$$f_6 = (\underline{конт}\, t, \underline{функ}\, x, y)\,\underline{функ} : (T:4, X, Y)\, f_4^0(f_4^0(X,X), f_4^0(X,X)) \,\big|$$
$$(T:3, X, Y)\, f_5^0(f_5^0(X,Y), f_5^0(X,Y)) \,\big|\, (T:1, X, Y)\, f_1^0(f_2^0(f_1^0(X), f_1^0(Y))) \,\big|$$
$$(T:0\,|\,2, X, Y) f_3^0(X,Y)$$
$$f_7 = (\underline{текст}\, x)\,\underline{функ} : (X) X$$
$$f_8 = (\underline{конт}\, t, \underline{текст}\, x)\,\underline{конт} : (M, 'И, НЕ')\, M:0 \,\big|\, (M, 'ИЛИ, НЕ')\, M:1 \,\big|$$
$$(M, 'И, ИЛИ, НЕ')\, M:2 (M, 'И - НЕ')\, M:3 \,\big|\, (M, 'ИЛИ - НЕ')\, M:4$$

Базисные функции f_j^0:

$$f_1^0 = (\underline{знач}\, x)\,\underline{знач} : (X) НЕ(X)$$
$$f_2^0 = (\underline{знач}\, x, y)\,\underline{знач} : (X,Y) ИЛИ(X,Y)$$
$$f_3^0 = (\underline{знач}\, x, y)\,\underline{знач} : (X,Y) И(X,Y)$$
$$f_4^0 = (\underline{знач}\, x, y)\,\underline{знач} : (X,Y) ИЛИ - НЕ(X,Y)$$
$$f_5^0 = (\underline{знач}\, x, y)\,\underline{знач} : (X,Y) И - НЕ(X,Y)$$

Память грамматики используется для записи базиса операций, который указывает пользователь в исходных данных. Память грамматики выглядит следующим образом:

$$M \to 0\,|1|\,2\,|3|\,4\,|\varepsilon\,,$$

где ε – пустое множество.

Таким образом, была составлена функциональная грамматика, способная при подаче на вход пользователем системы булевых функций провести простейшую декомпозицию и на выходе построить структурное описание комбинационной схемы в соответствии с указанным базисом операций.

2. Пример разбора логических выражений

В качестве примера рассмотрим простейшую реализацию системы булевых функций комбинационной схемой на логических элементах $И, ИЛИ, НЕ$.

$$P = \bar{a} \cdot \bar{b} \cdot \bar{c} + \bar{a} \cdot b \cdot c;$$
$$S = b \cdot \bar{c} + a \cdot c;$$

На вход функциональной грамматики должна быть подана следующая последовательность: система булевых функций, представленная символа-

ми алфавита грамматики и базис элементов, в котором необходимо построить структурное описание схемы.

$$P = \neg(a)\cdot\neg(b)\cdot\neg(c) + \neg(a)\cdot b\cdot c,\ S = b\cdot\neg(c) + a\cdot c;\ И, ИЛИ, НЕ.$$

На первом шаге работы заданная функциональная грамматика произведет синтаксический разбор входной информации (рис. 1, 2). Согласно правилам входная последовательность будет рассматриваться как два нетерминала $< СБФ >$ и $< Базис >$, разделенные символом «,». Первый нетерминал развернется в ветку разбора системы булевых функций, при этом в каждом узле разбора выставляя соответствующую функцию грамматики. Второй нетерминал развернется в текстовую последовательность, в нашем случае — $И, ИЛИ, НЕ$.

На втором этапе будет производиться выполнение выставленных во время разбора функций. Следует отметить, что функции, определяющие структурное описание схемы, не имеют доступа к расположенному в другой ветви разбора значению базиса элементов. Поэтому результатом выполнения функций будет синтаксически разобранная, но не выполненная суперпозиция. Таким образом, будет построена суперпозиция, описывающая логику работы проектируемого устройства, которая в качестве аргумента будет передана функции f_0. Вычисление значений функций будет произведено лишь после выполнении функция f_8, которая запишет в модель памяти грамматики значение соответствующего базиса.

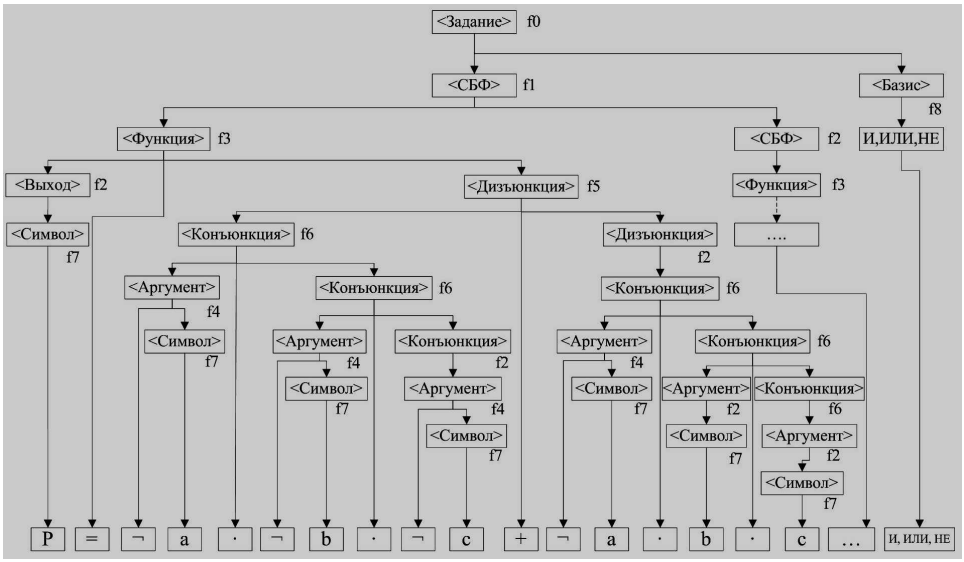

Рис. 1. Дерево синтаксического разбора функции P

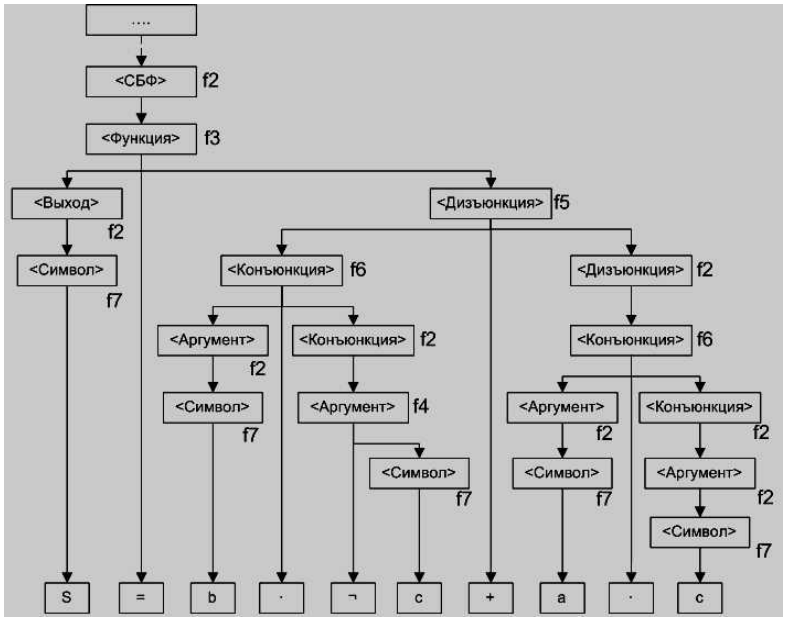

Рис. 2. Дерево синтаксического разбора функции S

Рассмотрим принцип работы функций на примере

$$f_1 = (\underline{функ}\, x, y)\, \underline{функ} : (X, Y) X, Y$$

Представленная функция состоит из двух частей: заголовка и тела. Заголовок, в свою очередь, так же разделен на две части: в скобках задаются типы и имена параметров, за скобками объявляется тип возвращаемого значения. Для функции f_1 объявлено два параметра, имеющие тип $\underline{функ}$, результатом будет так же тип $\underline{функ}$.

Каждому параметру и вырабатываемому результату функции должен быть указан один из четырех типов: $\underline{знач}$, $\underline{функ}$, $\underline{текст}$ или $\underline{конт}$ [8]. Если параметр имеет тип $\underline{знач}$, то при обращении аргумент будет рассматриваться как значение выполненной суперпозиции. Если $\underline{функ}$, то аргумент подается на вход в виде синтаксически разобранной, но не вычисленной суперпозиции. Если $\underline{текст}$, то аргумент рассматривается как последовательность символов. Если $\underline{конт}$, то значение аргумента берется из общей памяти.

Продолжим рассмотрение функции f_1, расположенной в верхней части дерева разбора (рис. 1). В теле функции в скобках указаны формальные параметры *X* и *Y*, которые в процессе вычисления будут отождествлены с фактическим аргументом. Поскольку в заголовке f_1 аргумент имеет тип

функ, то фактический параметр будет являться не выполненная суперпозиция нижестоящих функций. Результат работы f_1 будет следующим:

$$f_1(f_3(f_2(f_7(P)),f_5(f_6(f_4(f_7(a)),f_6(f_4(f_7(b)),f_2(f_4(f_7(c))))),f_2(f_6(f_4(f_7(a)),$$
$$f_6(f_2(f_7(b)),f_2(f_2(f_7(c)))))))),f_2(f_3(f_2(f_3(f_2(f_7(S)),f_5(f_6(f_2(f_7(b)),$$
$$f_2(f_4(f_7(c)))),f_2(f_6(f_2(f_7(a)),f_2(f_2(f_7(c))))))))))).$$

В ветви разбора базиса операций находится узел, которому соответствует функция:

$$f_8 = (\underline{конт}\, t, \underline{текст}\, x)\,\underline{конт}:(M,'И,НЕ')M:0\,|(M,'ИЛИ,НЕ')M:1\,|$$
$$(M,'И,ИЛИ,НЕ')M:2(M,'И-НЕ')M:3\,|(M,'ИЛИ-НЕ')M:4$$

Представленной функции в заголовке указаны два аргумента: $t\,\underline{конт}$ и $x\,\underline{текст}$. Фактическими параметрами функции будут текущее значение памяти M равное ε и текст $И, ИЛИ, НЕ$. Возвращаемым значением указан тип *конт*, обозначающий, что вычисленное значение функции будет записано в память M. В теле функции с помощью условных операторов производится выбор значения базиса. Результатом работы является число 2, записанное в память M.

Теперь рассмотрим следующую выше в дереве разбора функцию

$$f_0 = (функ\, x, знач\, y)\, знач:(X)\, eval\, X.$$

Она имеет два аргумента x и y, имеющие тип *функ* и *знач*. В теле функции используется лишь один формальный параметр, поэтому параметр y в процессе вычислений участвовать не будет. В теле функции используется операция *eval*, необходимая для вычисления полученной в качестве аргумента суперпозиции. Во время ее выполнения функции f_4, f_5, f_6 произведут обращение к памяти и в зависимости от полученного значения, в описание схемы будет добавлен узел из логических элементов $И, ИЛИ, НЕ$.

Таким образом, получим результат:

$$P(ИЛИ(И(НЕ(a),И(НЕ(b),НЕ(c))),И(НЕ(a),И(b,c)))),$$
$$S(ИЛИ(И(b,НЕ(c)),И(a,c)))$$

Данному описанию соответствует следующая комбинационная схема:

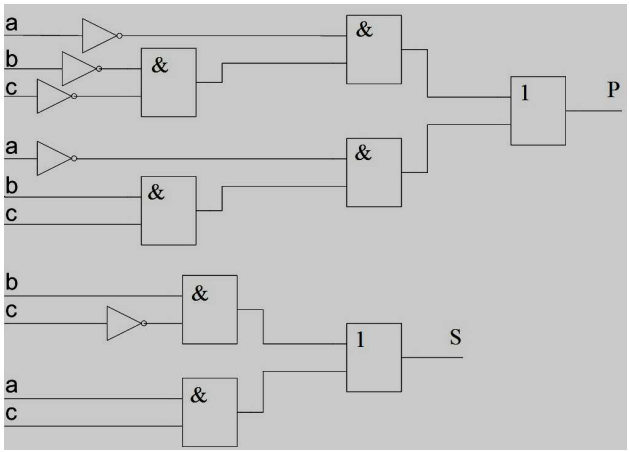

Рис. 3. Результат синтеза комбинационной схемы

Заключение

Предложенная функциональная грамматика осуществляет многоблочную последовательную разделительную декомпозицию булевых функций. Реализован механизм, позволяющий строить комбинационные схемы на различных базисах логических элементов. Данный механизм демонстрирует порождающую способность функциональной грамматики.

При усовершенствовании системы путем расширения списка базисов элементов, введением системы их преобразования, дополнительных функций можно реализовать как традиционные методы декомпозиции, так и решать задачи синтеза схем с учетом различных ограничений на структуру для программируемых логических интегральных схем. С использованием памяти функциональной грамматики данные ограничения могут управлять процессом вычисления суперпозиции функций.

Литература

1. Глушков В. М. Синтез цифровых автоматов. М.: Физматгиз, 1962. 476 с.

2. Майоров С. А., Новиков Г. И., Немолочнов О. Ф. Проектирование цифровых вычислительных машин: учеб. пособие для вузов. М.: Высшая школа, 1972. 344 с.

3. Perkowski M. A., Grygiel S. A Survey of Literature on Functional Decomposition. Version IV (Technical Report). Portland, USA: Portland State University, Department of Electrical Engineering, 1995. 188 p.

4. Соловьев В. В. Синтез комбинационных и последовательных схем на программируемых логических устройствах // Управляющие системы и машины. 1997. № 1/3. С. 66–71.

5. Басалин П. Д., Белокрылов П. Ю., Згурский Д. С. Синтез схем произвольной комбинационной логики в нейросетевом базисе с применением метода имитации отжига // Вестник ННГУ. 2008. № 5. С.126–130.

6. Кравченко В. А., Могнонов П. Б., Чимитов Д. Н. Представление знаний в функциональных грамматиках // Вестник СибГАУ. 2011. № 5 (38). С. 55–61.

7. Кравченко В. А., Ширапов Д. Ш. Построение баз знаний для решения задач методом функциональных грамматик // Вестник БГУ. 2015. № 9. С. 96–102.

8. Тузов В.А. Математическая модель языка. Л.: Изд-во Ленингр. ун-та, 1984. 176 с.

REALIZATION OF BOOLEAN FUNCTIONS BY COMBINATION CIRCUITS USING FUNCTIONAL GRAMMARS

Vasiliy V. Chimitov
Research Assistant,
East-Siberian State University of Technology and Management
40v Kluchevskaya, Ulan-Ude 670033, Russia
E-mail: cvh1304@yandex.ru

Petr B. Mognonov
Cand. Sci., A/Prof.,
East-Siberian State University of Technology and Management
40v Kluchevskaya, Ulan-Ude 670033, Russia
E-mail: pmognonov@mail.ru

In the article, we propose a method for decomposing Boolean functions using functional grammars. The main problems of constructing functional grammars for implementation of Boolean functions by combinational circuits are considered. The generative ability of functional grammars is demonstrated by the example of combination circuit synthesis.
Keywords: grammar; combination circuits; decomposition of Boolean functions.