УДК 681.32

УСОВЕРШЕНСТВОВАННЫЙ ЛОГИЧЕСКИЙ ЭЛЕМЕНТ FPGA

С. Ф. Тюрин, А. С. Прохоров

Пермский национальный исследовательский политехнический университет

Поступила в редакцию 11.10.2016 г.

Аннотация. Логические элементы программируемых логических интегральных схем (ПЛИС) типа FPGA (field-programmable gate array), часто обозначаемые как LUT (Look Up Table), вычисляют одну логическую функцию от, как правило, не более 4-х переменных. В статье предлагается новый логический элемент DubleLUT и оценивается его сложность в сравнении с реализацией двух функций в двух разных LUT. Делается вывод о выигрыше в сложности порядка 10 % без учёта сложности коммутаций переменных, что ещё увеличит выигрыш. Оценивается также выигрыш в реализации функций более 4-х переменных.
Ключевые слова: логический элемент, ПЛИС типа FPGA, LUT, транзистор.
Annotation. Logic elements programmable logic integrated circuits (FPGA) type FPGA (field-programmable gate array), so-called a LUT (Look Up Table), is calculated by a logic function is generally not more 4 variables. The article discusses the new logic element DubleLUT and estimated its complexity compared with the realization of two functions in two different LUTs. The conclusion is about the success in the complexity by 10% without taking into account the complexity of switching variables which increase this gain. It is estimated as a gain in the realization of functions with more than 4 variables.
Keywords: FPGA, logic functions, logic element, LUT, transistor.

ВВЕДЕНИЕ

Логические ячейки (Logic Cell) или логические элементы ЛЭ являются основой программируемых логических интегральных схем (ПЛИС) типа FPGA (field-programmable gate array) [1] – и представляют из себя постоянные запоминающие устройства ПЗУ (называемые также LUT – Look Up Table), реализованные на мультиплексоре, входы данных которого настраиваются константами. 4-LUT изображён на рис.1.

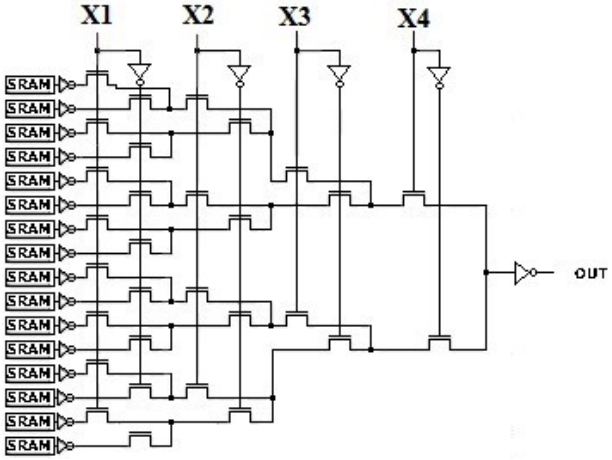

Рис. 1. LUT на четыре переменные (4-LUT)

4-LUT описывается выражением (1):

$$z_{out}(x_4x_3x_2x_1) = a\bar{x}_4\bar{x}_3\bar{x}_2\bar{x}_1 \vee b\bar{x}_4\bar{x}_3\bar{x}_2x_1 \vee$$
$$\vee c\bar{x}_4\bar{x}_3x_2\bar{x}_1 \vee d\bar{x}_4\bar{x}_3x_2x_1 \vee e\bar{x}_4x_3\bar{x}_2\bar{x}_1 \vee$$
$$\vee f\bar{x}_4\bar{x}_3x_2x_1 \vee g\bar{x}_4x_3x_2\bar{x}_1 \vee h\bar{x}_4x_3x_2x_1 \vee$$
$$\vee jx_4\bar{x}_3\bar{x}_2\bar{x}_1 \vee jx_4\bar{x}_3\bar{x}_2x_1 \vee kx_4\bar{x}_3x_2\bar{x}_1 \vee \quad (1)$$
$$\vee lx_4\bar{x}_3x_2x_1 \vee mx_4x_3\bar{x}_2\bar{x}_1 \vee nx_4x_3\bar{x}_2x_1 \vee$$
$$\vee ox_4x_3x_2\bar{x}_1 \vee px_4x_3x_2x_1$$

В ПЛИС Stratix III имеются адаптивные (перестраиваемые под требуемую задачу) логические блоки (ALM), которые объединяются в логические блоки (Logic Array Block, LAB) [2, 3], которые, как утверждается, реализуют функции даже 7 переменных. В силу ограничений Мида и Конвей на число последовательно соединённых транзисторов [4], дерево передающих транзисторов не может содержать более четырёх транзисторов в цепочке. То есть используется декомпозиция многоразрядного LUT на LUT меньшей раз-

рядности, то есть построение дерева из поддеревьев. Анализ выражения (1) позволяет установить, что логическая функция на конкретном наборе переменных реализуется половиной дерева транзисторов рис. 1, то есть вторая половина может быть использована для реализации второй функции, зависящей от тех же переменных. Рассмотрим особенности такого предлагаемого DubleLUT.

LUT ПЛИС FPGA, ВЫЧИСЛЯЮЩИЙ ДВЕ ЛОГИЧЕСКИЕ ФУНКЦИИ ОДНОВРЕМЕННО

В отличие от рис. 1 вводится дополнительная пара транзисторов и соответствующий инвертор с выходом OUT2. Вторая функция от тех же переменных X_4 X_3 X_2 X_1 настраивается второй группой конфигурационной памяти SRAM. При этом вводятся две группы отключающих две группы SRAM транзисторов. В случае активации одной половины дерева, соответствующая половина группы отключающих транзисторов подключается, а вторая половина отключается, обеспечивая подключение половины группы SRAM второй функции – рис. 2.

Причём таблица истинности для второй функции как бы «перевёрнута» относительно первой функции, но во второй паре транзисторов используется та же старшая переменная X_4.

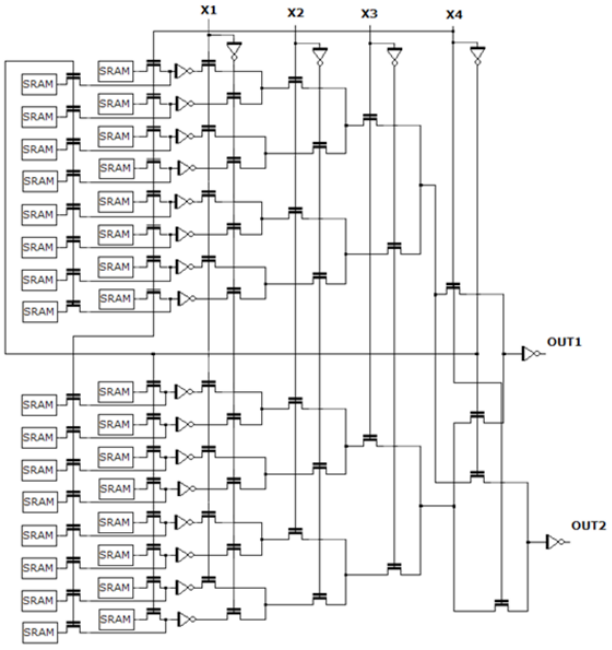

Рис. 2. DubleLUT на четыре переменные (Duble4-LUT)

ОЦЕНКА СЛОЖНОСТИ DUBLELUT

Сложность LUT (рис.1) с учётом сложности SRAM (6 транзисторов) и инверторов (2 транзистора) оценивается как ($1 \le n \le 4$):

$$(2+6)\cdot 2^n + (2^{n+1}-2) + 2n + 2 =$$
$$= 2^3 \cdot 2^n + 2^{n+1} + 2n = 5\cdot 2^{n+1} + 2n$$

Для вычисления двух функций используются два LUT, тогда при $n=4$ получим, что получаем без учёта сложности коммутаций переменных к двум LUT:

$$2\cdot(5\cdot 2^{n+1} + 2n) = 2\cdot(5\cdot 2^{4+1} + 2\cdot 4) = 336$$

В случае совмещения двух функций в предлагаемом DubleLUT (без учёта сложности коммутаций переменных):

$$(5\cdot 2^{n+1} + 2n) + (6+2)\cdot 2^n + 2 + 2 =$$
$$= (5\cdot 2^{n+1} + 2n) + 2^{n+3} + 4 =$$
$$= (5\cdot 2^{4+1} + 2\cdot 4) + 2^{4+3} + 4 = 300$$

То есть экономия в основном за счёт дерева транзисторов. Выигрыш чуть больше 10 %, да задержка увеличивается на один транзистор. То есть, например, можно вычислять и сумму, и перенос в полном LUT-сумматоре. Если учесть, что число логических элементов в настоящее время более миллиона, то выигрыш существенный. Диаграммы сложности известного и предлагаемого LUT в зависимости от *n* представлены на рис. 3.

При построении LUT более 4-х переменных необходима декомпозиция по 1-LUT, 2-LUT, 3-LUT или 4-LUT [2,3]. Пример LUT на пять переменные (5-LUT), построенного из двух 4-LUT и 1-LUT изображён на рис.4:

Инверторы по выходам 4-LUT убирать нельзя. В соответствие с ограничениями Мида и Конвей [4], они выполняют роль восстановителей сигнала, поэтому, так как их теперь нечётное число, настройка SRAM должна быть инверсной. 6-LUT из четырёх 4-LUT и одного 2-LUT изображён на рис. 5.

На рис. 5 входы 2-LUT имеют инверторы, поэтому, поскольку число инверторов на

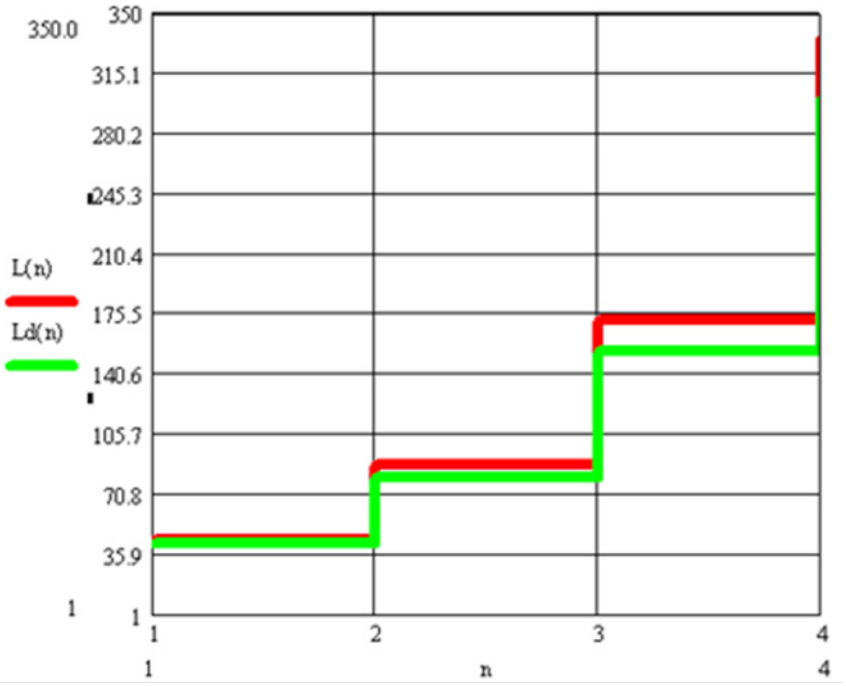

Рис. 3. Диаграммы сложности известного и предлагаемого LUT в зависимости от n =1, 2, 3, 4

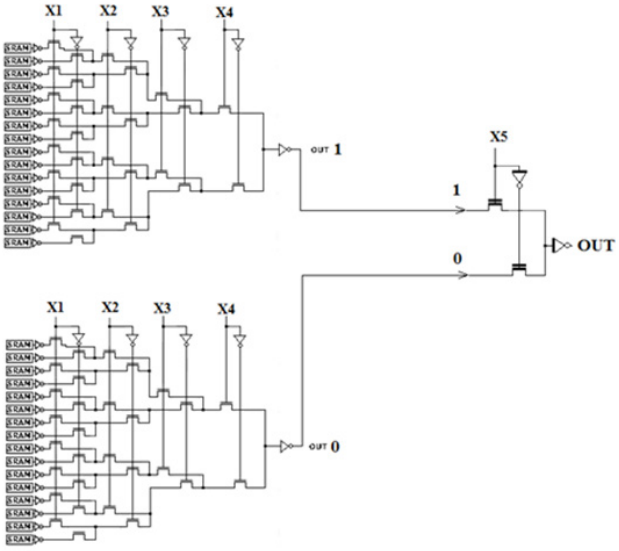

Рис. 4. 5-LUT на пять переменные из двух 4-LUT и 1-LUT

пути сигнала чётное, настройки записываются как обычно. Оценим сложность Duble LUT при использовании подобной декомпозиции.

ОЦЕНКА СЛОЖНОСТИ DUBLE LUT ПРИ ДЕКОМПОЗИЦИИ

Пусть k – это размерность основного (базового) LUT ($k \in \{1,2,3,4\}$). Причём для 1-LUT в принципе до $n=4$ нет необходимости в выходном инверторе. Больше 4 по указанным выше ограничениям k в настоящий момент пока не практикуется.

Оценим сложность LUT без декомпозиции («идеальная» сложность, поскольку такое может быть только до $n=4$, не более):

$$L_n = 2^n \cdot 8 + 2^{n+1} + 2n,$$

где $2^n \cdot 8$ – сложность настройки (по каждому входу настройки необходимо 6 транзисторов SRAM+2 транзистора – инвертор на входе дерева передающих транзисторов); 2^n – сложность инверторов по n переменным; 2^{n+1} – сложность дерева передающих транзисторов с выходным инвертором.

При декомпозиции n-дерева по k LUT, $k \in \{1,2,3,4\}$, $n \geq k$:

$$L_{n.k} = 2^n \cdot 8 + (2^{k+1} + 2k) \cdot 2^{n-k} +$$

$$+(2^{2^{n-k}+1} + 2^{n-k+1}) + 2n,$$

где 2^{k+1} – сложность дерева k LUT, $2k$ – число транзисторов в k инверторах, таких деревьев необходимо 2^{n-k}, для соединения получаемых при декомпозиции 2^{n-k} деревьев необходимы ещё LUT на 2^{n-k} входов (которые тоже можно декомпозировать), соответственно сложность $2^{n-k+1} + 2 \cdot 2^{n-k} = 2^{n-k+2}$, где 2^{n-k+1} – сложность дерева с выходным инвертором, $2 \cdot 2^{n-k} = 2^{n-k+2}$ – сложность входных инверторов. Тогда для реализации двух логических функций получим сложность:

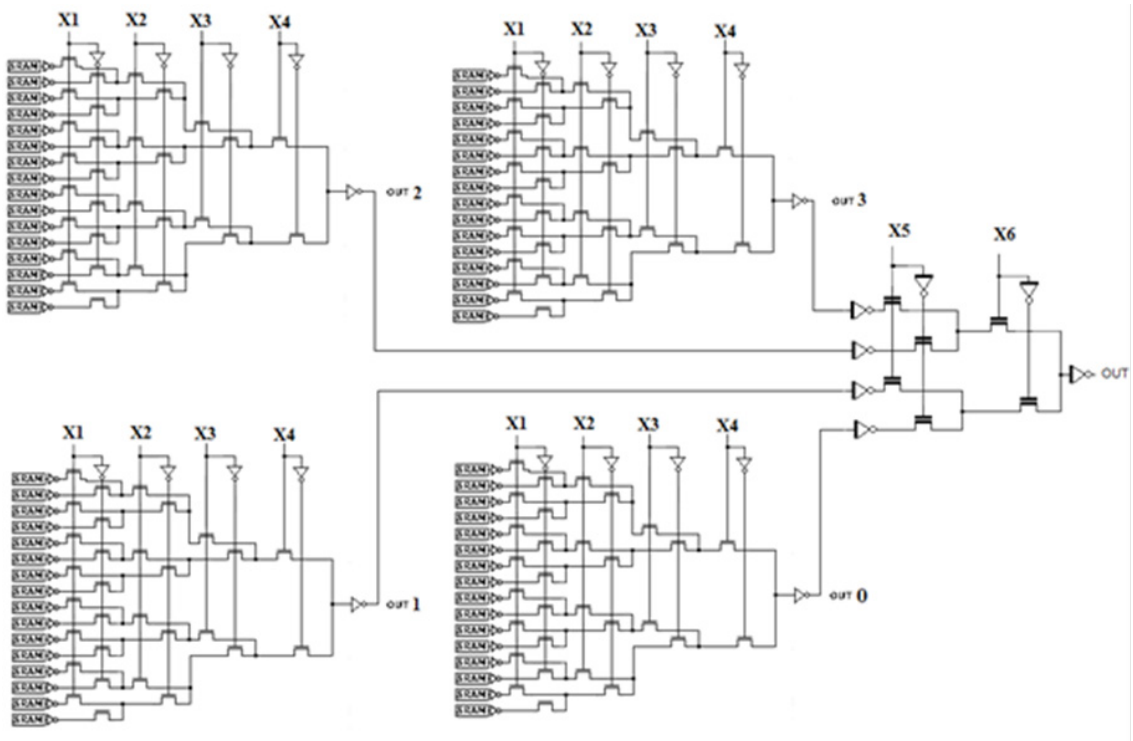

Рис. 5. 6-LUT из четырёх 4-LUT и одного 2-LUT

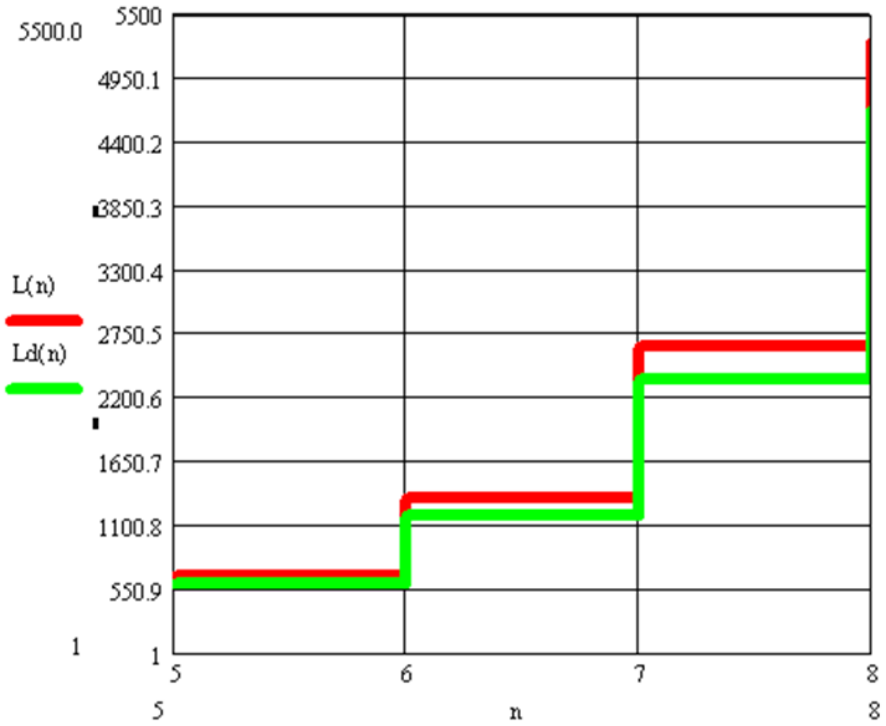

Рис. 6. Диаграммы сложности известного и предлагаемого LUT в случае декомпозиции в зависимости от n = 5, 6, 7, 8 при k = 4

$$L_{n.k} = 2[2^n \cdot 8 + (2^{k+1} + 2k) \cdot 2^{n-k} + \\ + (2^{2^{n-k}} + 1 + 2^{n-k+1}) + 2n].$$

С использованием Duble LUT будет:

$$L_{d-n.k} = [2^n \cdot 8 + (2^{k+1} + 2k) \cdot 2^{n-k} + \\ + (2^{2^{n-k}+1} + 2^{n-k+1}) + 2n] + 2^{n+3} + 4.$$

Диаграмма сложности представлена на рис. 6.

ВЫВОДЫ

Таким образом, использование Duble LUT позволяет получить выигрыш в сложности при реализации двух функций за счёт незначительного снижения быстродействия (задержка увеличивается на величину задержки одного передающего транзистора). Это позволяет увеличить функциональность LUT, например, в случае реализации суммирования. Однако вторая функция может зависеть только от тех же переменных, что и первая функция. Целесообразно в дальнейшем рассмотреть вопрос использования, предлагаемого Duble LUT в области цифровой обработки сигналов. По результатам исследования подана заявка на выдачу патента.

СПИСОК ЛИТЕРАТУРЫ

1. *Цыбин С.* Программируемая коммутация ПЛИС: взгляд изнутри. – Режим доступа: http://www.kit-e.ru/articles/plis/2010_11_56.php (дата обращения 16.12.2014)

2. *Золотуха Р., Комолов Д.* Stratix III – новое семейство FPGA фирмы Altera. – Режим доступа: http://kit-e.ru/assets/files/pdf/2006_12_30.pdf (дата обращения 28.11.2015)

3. Использование ресурсов ПЛИС Stratix III фирмы Altera при проектировани микропроцессорных ядер. – Режим доступа: http://www.kit-e.ru/articles/plis/2010_2_39.php (дата обращения: 27.11.2015).

4. *Ульман Дж. Д.* Вычислительные аспекты СБИС. Пер. с англ.: А. В. Неймана. Под ред. П.П. Пархоменко. – М. : Радио и связь, 1990. – 480 с.

5. *Тюрин С. Ф.* Функционально-полные толерантные булевы функции // Наука и технология в России. – № 4. – 1998. – С. 7–10.

6. *Тюрин С. Ф.* Синтез адаптируемой к отказам цифровой аппаратуры с резервированием базисных функций // Приборостроение. – 1999. – № 1. – С. 36–39.

7. *Тюрин С. Ф.* Адаптация к отказам одновыходных схем на генераторах функций с функционально-полными толерантными элементами // Приборостроение. – 1999. – № 7. – С. 32–34.

8. *Тюрин С. Ф.* Проблема сохранения функциональной полноты булевых функций при «отказах» аргументов // Автоматика и телемеханика. – 1999. – № 9. – С. 176–186.

9. *Тюрин С. Ф., Греков А. В., Громов О. А.* Определение функционально-полных толерантных булевых функций четырёх аргументов с учётом модели замыканий переменных // Доклады Академии военных наук. – № 5 (49). – Саратов. – 2011. – С. 35–44.

10. *Угрюмов Е. П.* Цифровая схемотехника : учебное пособие / Е. П. Угрюмов. – СПб : БХВ-Петербург, 2004. – 518 с.

Тюрин Сергей Феофентович – заслуженный изобретатель Российской Федерации, д-р техн. наук, профессор кафедры автоматики и телемеханики, электротехнический факультет, Пермский национальный исследовательский политехнический университет.
E-mail: tyurinsergfeo@yandex.ru

Прохоров Андрей Сергеевич – аспирант кафедры автоматики и телемеханики, электротехнический факультет, Пермский национальный исследовательский политехнический университет.
E-mail: прохор007@yandex.ru

Tyurin Sergey Feofentovich – Honored Inventor of the Russian Federation, Doctor of Technical Sciences, Professor at the Department of Automation and Telemechanics, Electrical Engineering Faculty, Perm National Research Polytechnic University.
E-mail: tyurinsergfeo@yandex.ru

Prokhorov Andrey Sergeevich – postgraduate student at the Department of Automation and Telemechanics, Electrical Engineering Faculty, Perm National Research Polytechnic University.
E-mail: прохор007@yandex.ru