УДК 004.056

КРЫЛОВА В.А., кандидат технических наук (Национальный технический университет «Харьковский политехнический институт»),
ДЕМИЧЕВ А.И., аспирант кафедры специализированных компьютерных систем (Украинский государственный университет железнодорожного транспорта),
МИРОШНИК А.Н., студент (Национальный технический университет «Харьковский политехнический институт»)

Исследование оптимальности реализации технологического картографирования на ПЛИС типа FPGA

Эта статья призвана определить оптимальность алгоритмов технического картографирования FPGA технологий. Разрабатывается алгоритм, основанный на технологии SAT (Boolean satis f ability), который позволяет преобразовать маленькую подсхему с минимально возможным использованием числа генераторов логических функций (LUTs - Look Up Table). Эта технология применена к маленьким частям схем, которые уже были преобразованы при помощи лучших алгоритмов картографирования FPGA .

В большинстве случаях, оптимальное преобразование (картографирование) подсхем позволило использовать меньшее количество LUT, по сравнению с исходным алгоритмом преобразования. Показывается, что для некоторых схем суммарное усовершенствование занимаемого пространства может достигать 67%.
***Ключевые слова**: распределенные сети, ПЛИС, FPGA, шифрование, защита информации, SAT, LUT.*

Постановка проблемы в общем виде и ее связь с важными научными или практическими задачами

FPGAs (программируемая вентильная матрица) – реконфигурируемые интегральные схемы, характеризующиеся множеством программируемых логических блоков, окруженных конфигурируемыми межсоединениями. Самые современные устройства FPGA содержат программируемые логические блоки, основанные на K-input просмотровых таблицах (K-LUT), каждый K - LUT содержит 2^K конфигурируемых бит, с помощью которых реализуется любая K-input функция [1]. Рис. 1 иллюстрирует общую структуру 2-х входового LUT.

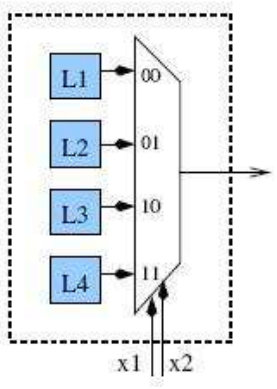

Рис. 1. Общая структура 2-х входового LUT

Число LUT-ов необходимое для реализации предоставленной схемы определяет размер и стоимость FPGA-based реализации. Поэтому одна из самых главных фаз автоматизированного проектирования FPGA – картографирование, которое переносит описание логической схемы на множество LUT элементов архитектуры FPGA.

Формулирование целей статьи (постановка задачи)

Цель технологии картографирования - сократить используемое пространство, задержку, или комбинацию того и другого в сети программируемых логических блоков. В этой работе оцениваются современные технологии алгоритмов нанесения на карту с точки зрения оптимизации занимаемого пространства. Картографирование по временным параметрам в этой статье рассматриваться не будет.

Изложение основного материала исследования. Методы исследования оптимальности реализации технологического картографирования

Процесс технологического картографирования и задача о покрытии.

Процесс технологического картографирования связывают с задачей о покрытии. Например, рассмотрим процесс составления схемы в LUT (представлен на рис. 2). Рисунок 2 (a) иллюстрирует начальную сеть на вентильном уровне, рисунок 2 (b) иллюстрирует возможное покрытие начальной сети с использованием 4-LUTs, и рис. 2 (c) иллюстрирует уже

покрытую сеть LUT. В предоставленной схеме, вентиль с меткой x покрывается обоими LUT, т.е. дублируется. Удивительно, но дублирование вентилей часто необходимо для минимизации области сетей LUT [1].

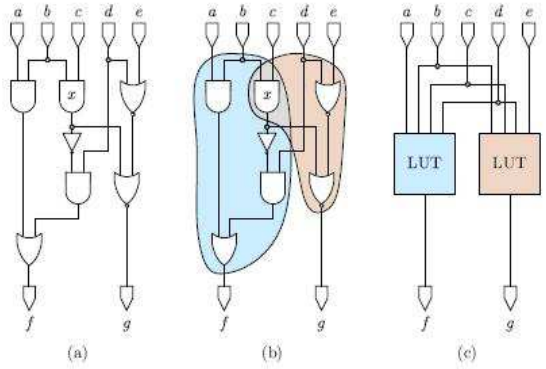

Рис. 2. Процесс составление схемы в LUT:
a) - начальная сеть на вентильном уровне,
b) - возможное покрытие начальной сети с использованием 4- LUTs , c) - покрытая сеть LUT

Фундаментальный вопрос этой статьи - насколько может быть уменьшена область для полученной сети LUT, созданной при помощи алгоритма технологического картографирования? Рассмотрим произвольную функцию f (i 0; i 1; : : : ; in) [2].

Предположим, что мы стремимся определить, можно ли уменьшить область при использовании трех или меньше двухвходовых LUT. Эта задача может быть решена при рассмотрении конфигурируемой виртуальной сети (Con fi gurable Virtual Network CVN) (рис. 3).

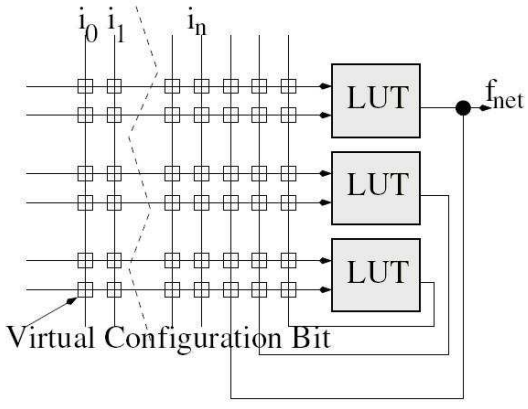

Рис. 3. Конфигурируемая виртуальная сеть

CVN состоит из входных линий соединенных с переменными i 0… in и трех 2-х входовых LUT . Поперечная связь позволяет входам LUT выбрать любую из входных или выходных линий другого LUT. Каждый “переключатель” на перекладине - конфигурируется виртуальным битом конфигурации. «0» - указывает, что контакт разъединен, а «1» - связь присутствует. Очевидно, что можно перечислить каждую возможную схемную конфигурацию, включающую в себя три 2-х входовых LUT, манипулируя виртуальными битами конфигурации для поперечной связи. Фактически, мы можем выразить выход fnet , как логическое выражение, включающее переменные i 0… in , виртуальные биты конфигурации V 1… Vm и конфигурационные биты таблицы истинности L 1… Lo . Получив эту формулу, встает вопрос: существуют ли значения V 1… Vm и L 1… Lo такие, что функция fnet (i 0; i 1; in ; V 1… Vm ; L 1… Lo) будет идентична функции f (i 0; i 1; in) для всех значений i 0… in ? Ответ на этот вопрос позволит дать технология SAT [3].

Получив логическое выражение в КНФ (CNF), где выражение состоит из произведения выражений и каждое выражение состоит из суммы переменных, ищем такое значение переменных таким образом, чтобы каждая переменная превращала выражение в “1” [4].

Сложность разрешения этой задачи экспоненциально возрастает с повышением числа переменных n виртуальной сети. Однако показано, что этот способ можно использовать только для повторного синтеза маленьких схем [5].

Получив оптимальный метод повторного синтеза для маленьких подсхем, мы без усилий многократно применяем эту технику к маленьким частям больших схем методом скользящего окна до тех пор, пока дальнейшее усовершенствование не может быть достигнуто [6].

Этот подход не гарантирует оптимальность при использовании больших схем, но это предоставляет нам некоторое указание свободной области при использовании первоначального картографирования [7 - 8].

Далее мы рассмотрим базовую литературу по управлению областью LUT, опишем оптимальный повторный синтез, основанный на SAT, опишем приложение из оптимального подхода повторного синтеза к сетям 4- LUT и предоставим набор результатов. Предоставим заключительные замечания и направления для будущей работы [9].

Повторный синтез области.

Проводя ресинтез области, мы должны взять LUT схему и попытаться сократить число генераторов логических функций LUTs в схеме с сохранением ее первоначального функционирования.

Чем больше LUTs можно исключить, тем больше первоначальная схема будет отличаться от оптимальной конфигурации. Мы упоминали, что

сокращение числа LUTs может быть достигнуто ресинтезом меньших подсхем с применением методики «скользящего окна» над большими схемами. Подсхемы, которые мы рассматриваем, имеют форму конуса. Поэтому ресинтез нескольких конусов должен сократить граф полной сети LUT.

Проблема повторного синтеза.

Определим, может ли n-возможный конус содержащий X N - LUTs быть повторно синтезирован в n -возможный конус содержащий X -1 N - LUTs или менее. Этого достигается формированием n-feasible FFC , содержащим меньшее количство LUTs, чем первоначальный конус, выражая FFC как логическое выражение КНФ. Потом, для КНФ выражения составляется таблица истинности переменных первоначального конуса. Если это выражение КНФ выполняется, повторный синтез к новому FFC возможен.

Для иллюстрации этого процесса рассмотрим рис. 4. Базовый конус (4 a) состоит из трех 2- LUTs которые реализуют функцию трех переменных. При такой реализации возможно повторно синтезировать 4 (a) в 4 (b), для сохранения одного LUT . Чтобы определить, возможен ли повторный синтез от 4 (a) к 4 (b) , 4 (b) должен быть превращен в выражение КНФ. Как сказано ранее, если выражение истинно, то повторный синтез может быть возможен.

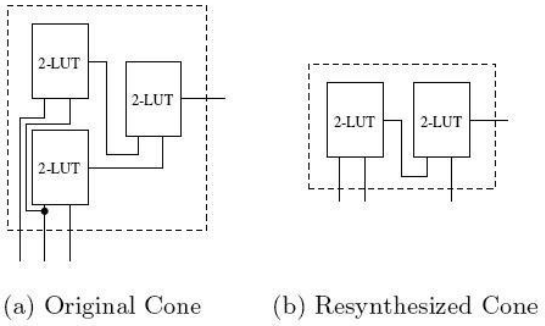

Рис. 4. Иллюстрация повторного синтеза

КНФ служит для выражения всех действительных векторов схемы. Например, рассмотрим рис. 5.

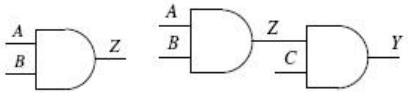

Рис. 5. Схемы

Выражение (1) будет удовлетворено, если и только если сигналы A , B , и Z , соответствуют функционированию AND вентиля (например (A = 0;B = X ; Z =0), (A = X ;B = 0; Z = 0) или (A = 1;B = 1; Z = 1)). Таким же образом, выражение (2) будет удовлетворяться только для действительных векторов, например (A = 1;B = 1; C = 0; Z = 1; Y = 0).

$$F_1(A,B,Z) = (A+\overline{Z})\cdot(B+\overline{Z})\cdot(\overline{A}+\overline{B}+Z) \quad (1)$$

$$F_2(A,B,C,Z,Y) = (A+\overline{Z})\cdot(B+\overline{Z})\cdot(\overline{A}+\overline{B}+Z)\cdot(Z+\overline{Y})\cdot(C+\overline{Y})\cdot(\overline{Z}+\overline{C}+Y) \quad (2)$$

Для проверки повторного синтеза, мы сначала формируем КНФ. Потом, мы накладываем ограничения на входные и выходные переменные в КНФ согласно таблице истинности конуса. Наконец, мы проверяем возможность присваивания, используя SAT решатель.

Например, давайте попытаемся спроецировать вторую функцию рис. 5 на 2x входовой конус. Полагаем, что вход C является конфигурационным битом. Чтобы проверить, что 2x входовая функция f(1;1)=1 выполняется, мы определяем, выполняется ли условие F 2(A = 1; B = 1; C ; Z ; Y = 1). Очевидно, это выполняется при C = 1. Однако это только показывает, что функция f (1; 1)= 1 действительна. Для проверки имеет ли место повторный синтез в схеме, выражение КНФ повторяется 2 n раз, чтобы сформировать новое выражение КНФ, где n представляет размер конуса. Каждая копия стандартной схемы КНФ представляет один куб в таблице истинности конуса. Опять SAT решатель выполняет проверку новой формулы КНФ на возможность повторного синтеза конуса.

Возвращаясь к базовому примеру LUT (рис. 4), рассмотрим следующие шаги, иллюстрирующие, как происходит преобразование. Рис. 6 - детальное представление рис. 4 (b) с внутренними проводами и конфигурацией вентилей КНФ.

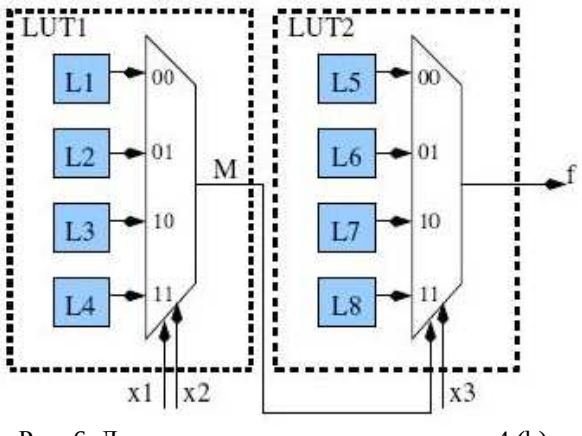

Рис. 6. Детальное представление рисунка 4 (b) с внутренними проводами и конфигурацией вентилей КНФ

В следующих шагах, f представляет собой функцию конуса для повторного синтеза, Xi представляет собой входной вектор x 1 x 2 x 3 = i , и fi=f(Xi).

Алгоритм

Шаг 1: Создание выражения КНФ для индивидуальных элементов в схеме повторного синтеза.

$$\begin{aligned} G_{LUT1} = \; & (x_1 + x_2 + \overline{L_1} + M) \cdot (x_1 + x_2 + L_1 + \overline{M}) \\ & \cdot (x_1 + \overline{x_2} + \overline{L_2} + M) \cdot (x_1 + \overline{x_2} + L_2 + \overline{M}) \\ & \cdot (\overline{x_1} + x_2 + \overline{L_3} + M) \cdot (\overline{x_1} + x_2 + L_3 + \overline{M}) \\ & \cdot (\overline{x_1} + \overline{x_2} + \overline{L_4} + M) \cdot (\overline{x_1} + \overline{x_2} + L_4 + \overline{M}) \end{aligned} \quad (3)$$

$$\begin{aligned} G_{LUT2} = \; & (x_3 + M + \overline{L_5} + f) \cdot (x_3 + M + L_5 + \overline{f}) \\ & \cdot (x_3 + \overline{M} + \overline{L_6} + f) \cdot (x_3 + \overline{M} + L_6 + \overline{f}) \\ & \cdot (\overline{x_3} + M + \overline{L_7} + f) \cdot (\overline{x_3} + M + L_7 + \overline{f}) \\ & \cdot (\overline{x_3} + \overline{M} + \overline{L_8} + f) \cdot (\overline{x_3} + \overline{M} + L_8 + \overline{f}) \end{aligned} \quad (4)$$

Шаг 2: Сформулируем схемное выражение КНФ из выражений (3) (4). Отметим, что выражение (5) зависит от входов схемы и выходов (x1, x2, x3, f), внутреннего провода (M), и переменных конфигурации (L 1- L8)

$$G_{resynth}(\boldsymbol{X_i}, f_i) = G_{LUT1} \cdot G_{LUT2} \quad (5)$$

Шаг 3: Копия выражения 5 и содержащия согласование входов и выходов для правильного функционирования функции f .

$$\begin{aligned} G_{\boldsymbol{Total}} = \; & G_{resynth}(\boldsymbol{X_0}, f_0) \cdot G_{resynth}(\boldsymbol{X_1}, f_1) \\ & \cdot G_{resynth}(\boldsymbol{X_2}, f_2) \cdot G_{resynth}(\boldsymbol{X_3}, f_3) \\ & \cdot G_{resynth}(\boldsymbol{X_4}, f_4) \cdot G_{resynth}(\boldsymbol{X_5}, f_5) \\ & \cdot G_{resynth}(\boldsymbol{X_6}, f_6) \cdot G_{resynth}(\boldsymbol{X_7}, f_7) \end{aligned} \quad (6)$$

В выражении (6), конфигурируемые биты представляются переменными (L 1-8) в каждом из Gresynth (Xi , fi), тогда как все другие сигналы представлены уникальными переменными в каждом случае. Это гарантирует, что только одна конфигурация будет существовать для всех кубов таблицы истинности. Наконец, выражение (6) передается в решатель SAT , который вернет истинное значение, если конус соответствует повторному синтезу.

Для простоты, в предыдущем примере мы проигнорировали гибкость трассировки FPGA , которые позволяет входам LUT перемещаться. Это чрезвычайно важно с тех пор как возросло число функций представленных в структуре повторного синтеза. Например, рис. 7 показывает, как функция от трех переменных может быть преобразована при помощи простой перестановки входов x $_1$ и x $_2$.

$x_1x_2x_3$	f
000	1
001	1
010	1
011	0
100	0
101	1
110	0
111	1

(a) $f(x_1, x_2, x_3)$

$x_1x_2x_3$	f
000	1
001	1
010	0
011	1
100	1
101	0
110	0
111	1

(b) $f(x_2, x_1, x_3)$, x_1 and x_2 swapped

Рис. 7. Функция трех переменных преобразована перестановкой входов

Распространив рис. 7 ко всем входным перестановкам, увеличивается число функций повторного синтеза представленных n !, где n - размер конуса повторного синтеза. Для того чтобы представить взаимозаменяемые входы в КНФ, добавлены виртуальные мультиплексоры (рис. 8). Они виртуальны в смысле, что их не существует в структуре повторного синтеза, но они служат для возможности изменять порядок входов в выражении КНФ.

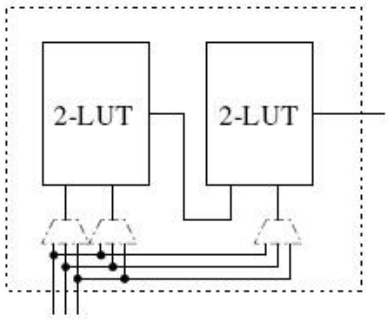

Рис. 8. Представление взаимозаменяемых входов в КНФ

Выводы

В статье определена оптимальность алгоритмов технического картографирования FPGA технологий. Разработан алгоритм, основанный на технологии SAT (Boolean satis f ability), который позволяет преобразовать маленькую подсхему с минимально возможным использованием числа генераторов логических функций (LUTs - *Look Up Table*). Многократно эта технология применялась к маленьким

частям схем, которые уже были преобразованы при помощи лучших алгоритмов картографирования FPGA.

Показано, что в большинстве случаях, оптимальное преобразование (картографирование) подсхем позволяло использовать меньшее количество LUT , по сравнению с исходным алгоритмом преобразования. Мы показываем, что для некоторых схем суммарное усовершенствование занимаемого пространства может достигать 67 %.

Литература

1. Miroshnik M. Application of software complex for query processing in the database management system with a view of dispatching problem solving in Grid systems. / Miroshnik M.A., Kotukh V.G., Selevko S.N. // Telecommunications and radio engineering. – 2013. Vol.27, № 10, P. 875-891.
2. Miroshnik M. Uses of programmable logic integrated circuits for implementations of data encryption standard and its experimental linear cryptanalysis. / Miroshnik M., Kovalenko M. // Інформаційно-керуючі системи на залізничному транспорті. – 2013. – №6, с.36-45.
3. Мирошник М.А. Проектирование компьютерных систем с интеллектуальной диагностической инфраструктурой. / М.А. Мирошник, В.Г. Котух, Э.Е. Герман // Радиотехника: Всеукраинский межведомственный научно-технический сборник.– Харків: ХНУРЕ, 2015. – Вип 180. – С. 64–67.
4. Miroshnik M. Implementation of cryptographic algorithms on FPGA-based digital distributed systems. / M. Miroshnik // Інформаційно-керуючі системи на залізничному транспорті: науково-технічний журнал. – Харків: УкрДУЗТ, 2015. – № 2 (111). – С. 25-30
5. Крылова В.А. Разработка методов оценки эффективности систем защиты информации в распределенных компьютерных системах / В.А. Крылова, А.Н. Мирошник // Інформаційно-керуючі системи на залізничному транспорті: науково-технічний журнал. – Харків: УкрДУЗТ, 2015. – № 2 (111). – С. 43-51.
6. Мірошник М.А. Розробка методів оцінки ефективності захисту інформації в розподілених комп'ютерних системах / М.А. Мірошник // Інформаційно-керуючі системи на залізничному транспорті: науково-технічний журнал. – Харків: УкрДУЗТ, 2015. – № 4 (113). – С. 39-43.
7. Мирошник М.А. Применение интеллектуальной диагностической инфраструктуры для управления кибербезопасностью. Часть 1 Интеллектуализация механизмов защиты. / М.А. Мирошник, В.А. Крылова, А.И. Демичев // Інформаційно-керуючі системи на залізнич-ному транспорті: науково-технічний журнал. – Харків: УкрДУЗТ, 2015. – № 6 (115). – С. 25-32.
8. Мирошник М.А. Применение интеллектуальной диагностической инфраструктуры для управления кибербезопасностью. Часть 2 Поддержка жизненного цикла системы киберзащиты. / М.А. Мирошник, В.А. Крылова, А.И. Демичев // Інформаційно-керуючі системи на залізнич-ному транспорті: науково-технічний журнал. – Харків: УкрДУЗТ, 2016. – № 1 (116). – С. 16-25.
9. Marina Miroshnyk Communication channel statistical characteristics research methods. / Marina Miroshnyk, Viktoriy Krulova. // Матеріали XIIIth Міжнародної конференції TCSET'2016 'Сучасні проблеми радіолектроніки, телекомунікацій, комп'ютерної інженерії (Львів-Славсько, Україна, 23 – 26 лютого 2016.). – Видавництво Львівської політехники. – Львів: Національний університет «Львівська політехніка», 2016. – С. 566-568.

Крилова В.А., Демічев О.І., Мірошник А.Н. Дослідження оптимальності реалізації технологічного картографування на ПЛІС типу FPGA. Ця стаття покликана визначити оптимальність алгоритмів технічного картографування FPGA технологій. Розробляється алгоритм, заснований на технології SAT (Boolean satis f ability), який дозволяє перетворити маленьку подсхему з мінімально можливим використанням числа генераторів логічних функцій (LUTs - Look Up Table). Ця технологія застосована до маленьких частин схем, які вже були перетворені за допомогою кращих алгоритмів картографування FPGA.

У більшості випадках, оптимальне перетворення (картографування) подсхем дозволило використовувати меншу кількість LUT, в порівнянні з вихідним алгоритмом перетворення. Показується, що для деяких схем сумарне удосконалення займаного простору може досягати 67%.

Ключові слова: розподілені мережі, ПЛІС, FPGA, шифрування, захист інформації, SAT, LUT.

Krylova V.A., Demichev A.I., Miroshnyk A.N. Technological FPGA type FPGA mapping implementation optimality research. This article aims to define FPGA technology technical mapping algorithms optimality. The algorithm is developed that based on SAT technology (Boolean satis f ability), which allows to convert a small sub-circuit with the lowest possible number of logic functions generators (LUTs - Look Up Table). This technology is applied to small schemes parts which have been converted using the best FPGA mapping algorithms.

In this paper we present a novel method for constructing arbitrarily large circuits that have known optimal solutions after technology mapping. Using these circuits and their derivatives (called LEKO and LEKU, respectively), we show that although leading FPGA technology mapping algorithms can produce close to optimal solutions, the results from the entire logic synthesis flow (logic optimization + mapping) are far from optimal. The best industrial and academic FPGA synthesis flows are around 140 times larger in terms of area on average, and in some cases as much as 500 times larger on LEKU examples. These results clearly indicate that there is much room for further research and improvement in FPGA synthesis.

In most cases, the optimal subcircuits transformation (mapping) makes possible to use a smaller number of the LUT, compared with the initial conversion algorithm. It is shown that for some schemes the total occupied space improvement can reach 67%.

Key words: distributed networks, FPGA, the FPGA, encryption, data protection, SAT, LUT.

Рецензент Листровой С.В., д.т.н., профессор, профессор кафедры СКС (УкрГУЖТ)

Поступила 01.03.2016 р.

Крилова В. А., *к.т.н., доцент кафедри автоматики і управління в технічних системах, Національний технічний університет «Харківський політехнічний інститут», Харків, Україна.*
Демичев А. И., *аспирант кафедри СКС, Український державний університет залізничного транспорту, Харків, Україна.*
Мірошник А. Н., *студент кафедри автоматики і управління в технічних системах, Національний технічний університет «Харківський політехнічний інститут», Харків, Україна.*

Krulova Viktoriy, *PhD, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine.*
Demichev Oleksandr*, post-graduate student, Ukrainian State University of Railway Transport, Kharkiv, Ukraine.*
Miroschnyk Anatoliy*, student, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine.*