МИКРОЭЛЕКТРОНИКА, 2022, том 51, № 1, с. 71–79

СХЕМОТЕХНИКА

УДК 004.415.2:621.3.049.771.14:519.17

ТЕОРЕТИКО-МНОЖЕСТВЕННЫЙ ПОДХОД К ПРЕДСТАВЛЕНИЮ ЭТАПА ТРАССИРОВКИ В МАРШРУТЕ ПРОЕКТИРОВАНИЯ В БАЗИСЕ ГЕТЕРОГЕННЫХ ПЛИС И РЕКОНФИГУРИРУЕМЫХ СнК

© 2022 г. В. И. Эннс[a], *, С. В. Гаврилов[b], **, М. А. Заплетина[b], ***

[a] *Научно-исследовательский институт молекулярной электроники (АО "НИИМЭ"), 1-ый Западный проезд, д. 12, стр. 1, Зеленоград, Москва, 124460 Россия*

[b] *Институт проблем проектирования в микроэлектронике Российской АН (ИППМ РАН), ул. Советская, д. 3, Зеленоград, Москва, 124365 Россия*

**E-mail: venns@niime.ru*

***E-mail: s.g@ippm.ru*

****E-mail: zapletina_m@ippm.ru*

Поступила в редакцию 27.05.2021 г.
После доработки 08.06.2021 г.
Принята к публикации 15.06.2021 г.

Представлен математический аппарат на основе теоретико-множественного подхода, позволяющий обеспечить оперативную настройку программных средств трассировки в рамках маршрута проектирования пользовательских схем в базисе гетерогенных программируемых логических интегральных схем и реконфигурируемых систем на кристалле. Разнообразие и сменяемость технологических и конструкторских решений, многообразие форм и способов программирования трассировочных ресурсов для реализации цепей коммутаций пользовательского проекта влекут за собой необходимость обеспечения быстрой настройки САПР на возможные изменения в архитектуре базового кристалла. С учетом этого требования, посредством приведенного математического аппарата в статье дано обобщенное описание трассировочных ресурсов гетерогенной ПЛИС и реконфигурируемой СнК, формализована задача трассировки цепей пользовательских проектных схем. Приведенные формальные описания также могут быть использованы при анализе пространства архитектурных решений в рамках маршрута проектирования проблемно-ориентированных гетерогенных ПЛИС и реконфигурируемых СнК.

Ключевые слова: трассировка, ПЛИС, СнК, проблемно-ориентированные ИС, теория графов, трассировочные ресурсы

DOI: 10.31857/S0544126922010069

1. ВВЕДЕНИЕ

Стандартные методы проектирования в базисе ПЛИС и реконфигурируемых систем на кристалле предполагают использование унифицированных трассировочных элементов, таких, как проходные ключи или ключи с буферами на выходе. Наличие в гетерогенных схемах сложно-функциональных блоков, блоков памяти, блоков логических элементов и др. приводит к необходимости применения различных вариантов трассировочных ресурсов, включающих, наряду с проходными ключами и буферами, более сложные элементы, реализующие логические функции инвертора, мультиплексора, буфера с третьим состоянием и пр. Это приводит к размыванию границы между этапами логического синтеза и топологического синтеза в классическом понимании этих задач. В этом случае этап топологического синтеза включает по необходимости решение задач логического ресинтеза [1] на этапе трассировки или после него, что требует разработки новых математических моделей и новой формализации задачи трассировки с элементами логического ресинтеза.

Разработка архитектуры проблемно-ориентированных гетерогенных программируемых логических интегральных схем и реконфигурируемых систем на кристалле представляет собой актуальную сложную задачу, решать которую нужно при условии жестких технологических ограничений и экономических требований [2–5]. Методам и моделям, используемым в процессе создания новых архитектур, посвящен ряд научных исследований [6–9]. В подходе, используемом в данной работе, при поиске архитектуры проблемно-ориентированных кристаллов ПЛИС и реконфигурируемых СнК предполагается учитывать конкретные зада-

чи и специфику целевого потребителя путем внедрения специального этапа программного прототипирования.

Программное прототипирование представляет собой новый этап в маршруте проектирования гетерогенных программируемых логических интегральных схем и реконфигурируемых систем на кристалле. Этот этап позволяет использовать набор типичных пользовательских проектных схем в качестве основы для определения базовых параметров архитектуры кристаллов от производителя. Реализация этого этапа на основе имеющихся программных средств проектирования на ПЛИС становится мотивацией для исследования и разработки новых математических моделей, методов и средств для автоматической настройки САПР на изменения архитектуры проектируемого базового кристалла с учетом потребностей конечного пользователя.

В данной статье представлен математический аппарат, основывающийся на теоретико-множественном подходе к обобщенному описанию пользовательского проекта и самого базового кристалла. В рамках введенных формальных обозначений поставлена задача трассировки в составе маршрута топологического проектирования в базисе гетерогенных ПЛИС и реконфигурируемых СнК. Приведенная математическая модель трассировочных ресурсов базового кристалла позволяет учесть широкий спектр способов программирования коммутационных элементов, а также разнообразие выполняемых ими логических функций. Кроме того, приведенное формальное описание коммутационных ресурсов базового кристалла и самой процедуры трассировки может быть использовано в рамках этапа программного прототипирования при анализе пространства архитектурных решений проблемно-ориентированных гетерогенных ПЛИС и реконфигурируемых СнК.

Далее, приведены основные обозначения и термины обобщенного описания базового проекта реконфигурируемой или программируемой гетерогенной системы и пользовательских проектируемых схем от конечного заказчика, функциональность которых требуется запрограммировать в кристалле от производителя. Предложено формальное описание коммутационных ресурсов базового кристалла, выполнена постановка задачи трассировки пользовательской схемы. Рассмотрены некоторые аспекты определения трассируемости пользовательских схем в базисе гетерогенной ПЛИС или реконфигурируемой СнК.

2. ОСНОВНЫЕ ОБОЗНАЧЕНИЯ И ТЕРМИНОЛОГИЯ ДЛЯ ОПИСАНИЯ БАЗОВОГО КРИСТАЛЛА И ПРОЕКТНОЙ СХЕМЫ

Обобщенное иерархическое описание проекта (как базового кристалла, так и пользовательской схемы) можно определить как триплет $\Pi = (S, L, s_m)$, где $S = \{s_i,\ i = 1,\ldots,|S|\}$ – множество схем в иерархическом описании проекта; $L \subset S$ – базис или подмножество базисных библиотечных подсхем для текущего уровня (или этапа) проектирования; $s_m \in S$, $s_m \notin L$ – главная схема или схема верхнего уровня.

Каждое из схемных описаний в иерархии проекта можно определить кортежем длины 5:

$$\forall s \in S\colon\ s = (\mu(s), E(s), N(s), P(s), C(s)),$$

где $\mu(s)$ – уникальное имя схемы (строка символов); $E(s) = \{e_i,\ i = 1,\ldots,|E(s)|\}$ – множество элементов в схеме; $N(s) = \{n_i,\ i = 1,\ldots,|N(s)|\}$ – множество цепей (электрических узлов) в схеме; $P(s) = \{p_i,\ i = 1,\ldots,|P(s)|\}$ – множество внешних выводов схемы; $C(s) = \{c_i,\ i = 1,\ldots,|C(s)|\}$ – множество соединений (коммутаций) схемы.

Множество элементов схемы может быть представлено как:

$$\forall e \in E(s)\colon\ e = (\mu(e), m(e), P(e)),$$

где $\mu(e)$ – уникальное имя элемента (строка символов); $m(e) \in S$ – модель элемента, представленная в иерархическом описании схемой более низкого уровня иерархии; $P(e) = \{p_i,\ i = 1,\ldots,|P(e)|\}$ – множество выводов элемента, совпадающее по составу с множеством внешних выводов модели, связанных с ним взаимно-однозначным соответствием

$$P(e) \leftrightarrow P(m(e)),\quad |P(e)| = |P(m(e))|.$$

Множество внешних выводов схемы описывается следующим образом:

$$\forall p \in P(s)\colon\ p = (\mu(p), \tau(p)),$$

где $\mu(p)$ – уникальное имя вывода; $\tau(p) \in \{\tau_{inp}, \tau_{out}, \tau_{bi}\}$ – тип вывода: вход, выход или двунаправленный.

Множество цепей схемы характеризуется именем $\forall n \in N(s)$: $\mu(n)$ и соответствующим цепи набором соединений $C(s)$, определяемым как подмножество пар:

$$C(s) = \left\{(p, n)\colon\ p \in \left(\bigcup_{i=1,\ldots,|E(s)|} P(e_i) \cup P(s)\right),\quad n \in N(s)\right\}.$$

Для любого контакта цепь единственная или не существует вовсе, таким образом, множество соединений в схеме определяется как однозначное отображение:

$$C^*(s) = \left\{\left(\bigcup_{i=1,\ldots,|E(s)|} P(e_i) \cup P(s)\right) \to (N(s) \cup \emptyset)\right\}.$$

При этом обратное отображение определяет собственно список соединений конкретной цепи и не может быть однозначным – количество соединений у каждой цепи должно быть не менее двух, в противном случае цепь будет считаться ошибочной или ложной:

$$C^{*-1}(s) = \left\{N(s) \to \left(\bigcup_{i=1,\ldots,|E(s)|} P(e_i) \cup P(s)\right)\right\}.$$

$$\forall n \in N(s): \left|\{(p,n): (p,n) \in C(s)\}\right| \geq 2.$$

Как правило, внешний вывод может быть у цепи только один:

$$\forall n \in N(s): \left|\{(p,n): (p,n) \in C(s) \wedge p \in P(s)\}\right| \leq 1.$$

Для текущего этапа проектирования подсхемы базисного библиотечного уровня не содержат внутренних данных и представляют собой “черные ящики”:

$$\forall s \in L: E(s) = \emptyset, \quad N(s) = \emptyset, \quad C(s) = \emptyset.$$

В этом случае моделирование подсхем нижнего уровня выполняется на основе встроенных моделей, и само описание “черных ящиков” может быть скрыто от внешнего пользователя. Например, на схемотехническом уровне проектирования к базисному библиотечному уровню относятся транзисторы, емкости, сопротивления, индуктивности и др.

В библиотеке L базового проекта следует выделить логические элементы L_{LE}, периферийные элементы ввода-вывода L_{IO}, сложно-функциональные макроблоки L_M, трассировочные элементы L_{Ro} и иные вспомогательные элементы L_{BB}, не содержащие в своем составе элементов перечисленных типов L_{LE}, L_{IO}, L_M, L_{Ro}, выполняющие вспомогательные функции (например, для программирования памяти), не связанные с непосредственным отображением элементов пользовательской проектируемой схемы:

$$L = L_{LE} \cup L_{IO} \cup L_M \cup L_{Ro} \cup L_{BB}.$$

Иерархическое описание проекта может быть преобразовано в соответствующее ему “плоское” представление путем рекурсивной процедуры раскрытия. Для заданного проекта $\Pi = (S, L, s_m)$ в “плоском” представлении целесообразно сохранить только те подсхемы, которые фактически применялись в главной схеме s_m. Обозначим через $\varphi(s, s_t)$ логическую функцию, определенную на Декартовом произведении $S \times S$, принимающую значение 1 тогда и только тогда, когда s фактически используется в s_t:

$$\varphi: S \times S \to \mathcal{B}; \quad \mathcal{B} = \{0,1\};$$

$$\varphi(s, s_t) = \left((s = s_t) \vee \left(\bigvee_{e \in E(s_t)} \varphi(s, m(e))\right)\right),$$

т.е.

$$\varphi(s, s_t) = 1, \quad \text{если} (s = s_t) \text{ или } \exists e \in E(s_t): \varphi(s, m(e)) = 1.$$

Таким образом, для заданного $\Pi = (S, L, s_m)$ проекта “плоское” $\Pi_f(\Pi) = (S_f, L_f, s_f)$ представление можно построить по следующим правилам:

$$L_f = \{s: (s \in L) \wedge \varphi(s, s_m)\};$$

$$S_f = \{s_f \cup L_f\};$$

$$s_f = (\mu(s_f), E(s_f), N(s_f), P(s_f), C(s_f), \quad \text{где } \mu(s_f) = \mu(s_m) \text{ и } P(s_f) \leftrightarrow P(s_m).$$

Имена элементов $\mu(e), e \in E(s_f)$ и имена цепей $\mu(n), n \in N(s_f)$ в “плоском” представлении должны быть уникальными и содержать информацию об именах элементов более высоких уровней иерархии, в состав которых входили подсхемы, содержащие рассматриваемый элемент, до раскрытия иерархии. Это достигается, например, за счет конкатенации имен элементов иерархии с использованием уникального разделителя.

3. ТЕОРЕТИКО-ГРАФОВАЯ МОДЕЛЬ ОПИСАНИЯ КОММУТАЦИОННЫХ РЕСУРСОВ ГЕТЕРОГЕННОЙ ПРОГРАММИРУЕМОЙ ИС

В соответствии с введенной терминологией для отображения цепей и коммутаций пользовательского проекта применяются элементы подсхем базового проекта e: $m(e) \in L_{Ro}$. Как и схемы библиотечного уровня $s \in L_{LE} \cup L_{IO} \cup L_M = L \backslash \{L_{BB} \cup L_{Ro}\}$, элементы трассировки могут программироваться. Принципиальное отличие состоит в том, что библиотечные элементы могут быть запрограммированы и характеризованы однократно для различных вариантов пользовательских схем. В то же время трассировочные элементы программируются индивидуально под конкретный набор коммутаций пользовательского проекта:

$$C^{*-1}(s_{mu}) = \left\{ N(s_{mu}) \to \left(\bigcup_{i=1,\ldots,\left|E(s_{mu})\right|} P(e_i) \right) \right\},$$
$$\forall n \in N(s_{mu}): \left| \{(p,n): (p,n) \in C(s_{mu})\} \right| \geq 2.$$

Предполагается, что к началу процедуры трассировки решена задача размещения, т.е. установлено отображение, в котором каждому элементу пользовательской схемы поставлен в соответствие элемент базового проекта (кристалла от производителя):

$$E(s_{mu}) \to \{e: e \in E(s_f), \;\; m(e) \in \{L_{LE} \cup L_M \cup L_{IO}\}\}.$$

При этом элементы $s_u \in L_u$, $s_u = (\mu(s_u), \emptyset, \emptyset, P(s_u), \emptyset)$ пользовательской библиотеки L_u проекта $\Pi_u = (S_u, L_u, s_{mu})$ реализованы путем установки следующих соответствий для выводов библиотечных схем базового кристалла $P(s) = \{p_i, \;\; i = 1,\ldots,|P(s)|\}$, $s \in L_{LE} \cup L_{IO} \cup L_M$,

$$P_r(s) \to P(s_u) \cup \{P_0, P_1, P_z\},$$

где P_0, P_1, P_z – условные обозначения вводов, предполагающие внешние соединения с узлом земли, питания или висячим узлом, соответственно. При этом обратное отображение $P(s_u) \to P_r(s)$ не обязательно однозначно и может отражать различные варианты внешних подключений.

Следовательно, не только каждому элементу пользовательской схемы поставлен в соответствие элемент базового проекта, но и каждому контакту (выводу) библиотечного элемента пользовательского проекта поставлены в соответствие контакты $P(e_i)$ библиотечных элементов базового проекта $e_i \in E(s_f)$, и, следовательно, узлы базового проекта:

$$C^{*-1}(s_{mu}) = \left\{ N(s_{mu}) \to \left(\bigcup_{i=1,\ldots,\left|E(s_f)\right|} P(e_i) \right) \right\}.$$

Таким образом, задача трассировки для реализации пользовательского проекта сводится к программированию проводящих путей, состоящих из трассировочных элементов, между выводами элементов и узлами базового проекта.

По аналогии со схемами библиотечного уровня, множество внешних выводов трассировочных элементов $P(s) = \{p_i, \;\; i = 1,\ldots,|P(s)|\}$, $s \in L_{Ro}$ можно разделить на два независимых подмножества по функциональному назначению: $P(s) = P_r(s) \cup P_m(s)$, где $P_r(s)$ – подмножество сигнальных или трассировочных выводов для соединения трассировочных элементов между собой и с выводами библиотечных элементов базового проекта; $P_m(s)$ – подмножество программируемых выводов для управления проводимостью трассировочных элементов путем назначения логических элементов программируемой памяти: $P_m(s) \to \mathcal{B}^{|P_m(s)|}$, $\mathcal{B} = \{0,1\}$. Кроме того, среди сигнальных или трассировочных выводов $p \in P_r(s)$ следует различать выводы по направлению распространения сигнала: $\tau(p) \in \{\tau_{inp}, \tau_{out}, \tau_{bi}\}$ – вход, выход или двунаправленный.

Введем следующие обозначения:

пусть $\nu(p)$, $p \in P_r(s) \cup P_m(s)$ – логическое значение на выводе p, т.е. $\nu: P(s) \to \mathcal{B}$, $\mathcal{B} = \{0,1\}$;

$c_i = \nu(p)$, $p \in P_m(s)$ – логическое значение управляющего сигнала;

$y_i = \nu(p)$, $p \in P_r(s)$, $\tau(p) \in \{\tau_{out}, \tau_{bi}\}$ – логическое значение входа;

$x_i = \nu(p)$, $p \in P_r(s)$, $\tau(p) \in \{\tau_{inp}, \tau_{bi}\}$ – логическое значение входа;

$\phi(c_i)$ – логическая функция управления открыванием трассировочного элемента, принимающая значение 1 для открытого состояния, 0 – для закрытого;

$\chi(x_i)$ – не константная логическая функция выхода в терминах входа для открытого состояния трассировочного элемента (в унарном варианте это инверсия или эквивалентность);

$\rho_i(c_i, y_i, x_i): \mathcal{B} \times \mathcal{B} \times \mathcal{B} \to \mathcal{B}$, $\mathcal{B} = \{0,1\}$ – характеристическая функция проводимости от x_i к y_i, принимающая значение 1 при корректных значениях и 0 в противном случае.

Корректными трассировочными элементами будем считать те, для которых совокупность характеристических функций проводимости может быть определена в импликативной форме (в форме конъюнкции импликаций) в следующем виде:

$$\rho(c, y, x) \equiv \bigwedge_i \rho_i(c_i, y_i, x_i) \equiv \bigwedge_i (\phi_i(c_i) \Rightarrow (y_i = \chi_i(x_i))),$$

где $\Rightarrow$ – символ операции импликации.

Среди различных вариантов корректных трассировочных элементов можно выделить следующие:

– проходной ключ: $\rho(g, s, d) \equiv \rho(g, d, s) \equiv (g \Rightarrow s = d)$;

– буфер: $\rho(\emptyset, y, x) \equiv (1 \Rightarrow y = x)$;

– инвертор: $\rho(\emptyset, y, x) \equiv (1 \Rightarrow y = \sim x)$;

– мультиплексор без инверсии:

$\rho(c, y, x) \equiv (\sim c \Rightarrow y = x_1) \wedge (c \Rightarrow y = x_2)$, $x = (x_1, x_2)^T$;

– мультиплексор с инверсией:

$\rho(c, y, x) \equiv (\sim c \Rightarrow y = \sim x_1) \wedge (c \Rightarrow y = \sim x_2)$, $x = (x_1, x_2)^T$;

– буфер с третьим состоянием: $\rho(c, y, x) \equiv (c \Rightarrow y = \sim x)$;

– резистор: $\rho(\emptyset, m, p) \equiv \rho(\emptyset, p, m) \equiv (1 \Rightarrow p = m)$.

Совокупность корректных трассировочных элементов базового проекта $\Pi_f(\Pi) = (S_f, L_f, s_f)$, $s_f = (\mu(s_f), E(s_f), N(s_f), P(s_f), C(s_f))$ может быть преобразована в трассировочный смешанный граф или ориентированный граф (с преобразованием каждого из двунаправленных ребер в пару дуг противоположного направления) $G_{Ro}(\Pi) = (V_{Ro}, A_{Ro})$ по правилам, представленным далее.

Множество вершин V_{Ro} графа $G_{Ro}(\Pi)$ взаимно однозначно соответствует подмножеству таких и только таких узлов $n \in N(s_f)$, для которых существует хотя бы одно соединение с элементом из подмножества трассировочных элементов:

$$\{n: n \in N(s_f) \wedge ((\exists e \in E(s_f): m(e) \in L_{Ro}) \wedge (\exists p \in P(e): (p, n) \in C(s_f)),$$
$$\Leftrightarrow \{v(n): v(n) \in V_{Ro}\}.$$

В качестве замечания необходимо отметить: предполагается, что в конструкции кристалла отсутствуют такие варианты, когда выводы логических элементов, макроблоков или периферийных элементов соединены друг с другом напрямую без трассировочных элементов.

Множество дуг графа $G_{Ro}(\Pi)$ формируется из импликативных форм характеристических функций проводимости трассировочных элементов $(\phi_i(c_i) \Rightarrow (y_i = \chi_i(x_i)))$. Для каждой пары $(y_i = \chi_i(x_i))$ создается новая дуга $a_{xy} = (v(x), v(y))$ между вершинами, соответствующими узлам $x \in N(s_f)$,

$y \in N(s_f)$, имеющим соединения с соответствующими выводами трассировочных элементов. При этом дуги $a_{xy} = (v(x), v(y))$ наследуют следующие характеристики трассировочного элемента:

$\phi(a_{xy})$ – логическую функцию управления открыванием трассировочного элемента, принимающую значение 1 для открытого состояния дуги $a_{xy} = (v(x), v(y))$, 0 – для закрытого;

$\chi(a_{xy})$ – логическую функцию состояния вершины, для которой дуга a_{xy} трассировочного графа является входящей, заданную в терминах состояния вершин, для которых дуга a_{xy} графа является исходящей, при открытом состоянии трассировочного элемента.

Таким образом, задача трассировки для реализации пользовательского проекта сводится к нахождению путей в трассировочном графе $G_{Ro}(\Pi) = (V_{Ro}, A_{Ro})$, соединяющих его вершины в соответствии с набором коммутаций узлов пользовательской схемы $n \in N(s_{mu})$:

$$C^{*-1}(s_{mu}) = \left\{ N(s_{mu}) \to \left(\bigcup_{i=1,\ldots,|E(s_f)|} P(e_i) \right) \right\}.$$

При этом направление искомых путей распространения сигнала определяется типом выводов коммутируемых элементов: от выходов $p_x \in P(e_i), \tau(p_x) \in \{\tau_{out}, \tau_{bi}\}$ к входам $p_y \in P(e_i)$, $\tau(p_y) \in \{\tau_{inp}, \tau_{bi}\}$.

Предположим, что $\tau = \{v_0, v_1, \ldots, v_n\}$, $v_i \in V_{Ro}$ – один из таких путей (маршрутов) с дугами $a_i = (v_i, v_{i+1})$, $i = 0,1,\ldots(n-1)$, $a_i \in A_{Ro}$, тогда:

$\phi(\tau) = \bigwedge_{n-1}^{i=0} \phi_i(a_i)$ – логическая функция управления открыванием трассировочного пути, определяющая назначение соответствующих элементов программируемой памяти, а

$\chi(\tau) = \chi_{n-1} \circ \chi_{n-2} \circ \ldots \circ \chi_0 = \chi_{n-1}(\chi_{n-2}(\ldots(\chi_0(v_0))))$ – логическая функция состояния концевой вершины пути – композиция (суперпозиция) соответствующих функций передачи сигнала для последовательности дуг.

В унарном варианте не константная логическая функция выхода в терминах входа – это отрицание (инверсия) или эквивалентность. В этом случае конечное значение $\chi(\tau)$ определяется подсчетом четности инверсных элементов в пути. При этом результирующая функция – отрицание при нечетном количестве отрицаний в пути и эквивалентность – при четном. Само по себе изменение логической функции пути с эквивалентности на любую другую требует переопределения (ресинтеза) логического элемента, соединенного с узлом схемы, соответствующим конечной вершине пути v_n по результатам трассировки.

Представленная модель описания коммутационных ресурсов и формализация задачи трассировки в рамках маршрута проектирования в базисе гетерогенных ПЛИС и реконфигурируемых СнК позволяет учесть широкий спектр способов программирования коммутационных элементов, а также разнообразие выполняемых ими логических функций.

4. ОЦЕНКА ТРАССИРУЕМОСТИ ПОЛЬЗОВАТЕЛЬСКОЙ СХЕМЫ В БАЗИСЕ ГЕТЕРОГЕННОЙ ПЛИС ИЛИ РЕКОНФИГУРИРУЕМОЙ СнК

В рамках программного прототипирования, наряду со стандартным контролем параметров пользовательской схемы по отношению к параметрам базового кристалла, решается обратная задача, заключающаяся в определении значений ключевых параметров архитектуры базового кристалла, необходимых для эффективной реализации заданного набора пользовательских схем. В этом случае этап трассировки предназначен для определения степени разводимости набора пользовательских схем в базисе прототипа разрабатываемого базового кристалла и формирования ограничений на его трассировочные ресурсы.

Основополагающей идеей программного прототипирования является анализ наборов пользовательских схем для определения требований и спецификаций разрабатываемого базового кристалла. Для задачи анализа пользовательских схем посредством автоматически настроенного САПР поле ПЛИС может быть представлено двумя способами: упрощенно в виде плоской матрицы логических элементов заранее определенного вида (например, 4-входовой LUT-элемент, реализующий таблицу истинности, с триггером); в виде двухуровневого блочного представления. Как правило, потенциал двухуровневого представления наилучшим образом раскрывается для описания островного типа архитектуры. В этом случае верхний уровень архитектуры кристалла может быть представлен в виде матрицы блоков LAB [10] логических элементов.

Среди характеристик пользовательских схем и базового кристалла от производителя можно выделить такие, соотношение между которыми наиболее существенным образом влияет на течение и исход процедуры трассировки. Характеристики пользовательских схем, как правило, рассчитываются динамически на этапах размещения, глобальной трассировки (в случае островной архитектуры целевого кристалла) или в процессе окончательной трассировки.

Перечислим основные характеристики пользовательских схем:

– требуемое количество элементарных (одноразрядных) связей между двумя логическими элементами или кластерами элементов (при превышении требуемого числа связей над фактически имеющимися в базовом кристалле в N раз, оценка трассируемости такой схемы будет различаться [16]), а также требуемая пропускная способность [11] канала (или переключательной коробки), требуемая связность переключательной коробки с каналами в маршруте для островной архитектуры;

– требуемая пропускная способность сечения по вертикали для заданного X, требуемая пропускная способность сечения по горизонтали для заданного Y (соответствующая характеристика базового кристалла – *фактическая* пропускная способность вертикальных и горизонтальных сечений);

– ограничение на задержку критического пути (частоту работы устройства);

– степень ветвления цепей пользовательской схемы (может быть использована для быстрой сортировки списка проектных соединений [12]);

– степень перегруженности отдельных трассировочных элементов, а также каналов для островной архитектуры, возникающая в процессе трассировки пользовательской схемы на каждом из этапов трассировки [13, 14], или как предварительная оценка результатов трассировки в целом на этапах глобальной трассировки [15] и размещения;

– остаточный путь до приемника, рассчитываемый динамически в процессе трассировки для ускорения трассировки за счет направленности поиска пути [12].

Выведем соотношение между требуемой для пользовательских схем и фактической пропускной способностью базового кристалла. Пусть задано двухуровневое блочное представление базового кристалла $\Pi_b(\Pi) = (S_b, L_b, s_b)$, содержащее следующие компоненты:

$$L_b = \{s: (s \in L) \wedge \varphi(s, s_m)\},$$
$$S_b = \{s_b \cup L_b \cup B\}, \quad L_b \cap B = \emptyset;$$
$$s_b = (\mu(s_b), E(s_b), N(s_b), P(s_b), C(s_b)), \quad \text{где:}$$
$$\mu(s_b) = \mu(s_m); \quad P(s_b) = P(s_m).$$

Рассматривается пользовательская схема $\Pi_u = (S_u, L_u, s_{mu})$, где $s_{mu} = (\mu(s_{mu}), E(s_{mu}), N(s_{mu}), P(s_{mu}), C(s_{mu}))$. Применительно к кластерному разбиению пользовательской схемы среди ее соединений (цепей $N(s_{mu})$), помимо внутренних и внешних, можно выделить смежные цепи, которые связаны с элементами как внутри, так и вне кластера (также с внешними выводами схемы):

$$N_{adj}(K_i) \subset N(s_{mu})$$
$$\forall n \in N_{adj}(K_i): \exists(p_1, n) \in C(s_{mu}), \exists(p_2, n) \in C(s_{mu}),$$
$$p_1 \in P(e_1) \wedge e_1 \in K_i \wedge (p_2 \in P(s_{mu}) \vee p_2 \in P(e_2) \wedge e_2 \notin K_i).$$

Тогда количество внешних выводов кластера или его *степень* можно определить как количество смежных цепей (узлов) $d(K_i) = |N_{adj}(K_i)|$.

Количество выходов или *исходящую степень* кластера $d^-(K_i)$ можно определить как количество смежных цепей (узлов), источником которых является элемент *внутри* кластера:

$$d^-(K_i) = |N^-(K_i)|, \quad N^-(K_i) \subset N_{adj}(K_i),$$
$$\forall n \in N^-(K_i): \exists(p, n) \in C(s_{mu}): p \in P(e) \wedge e \in K_i \wedge \tau(p) = \tau_{out}.$$

• Аналогично можно определить количество входов или *входящую степень* кластера $d^+(K_i)$ как количество смежных цепей (узлов), источником которых является элемент *вне* кластера или внешний вход схемы. В этом случае, количество входов/выходов кластера пользовательской схемы должно соответствовать количеству входов/выходов блока базового кристалла:

$$d^+(K_i) \le d^+(b) + d^*(b) = \left|P_r^+(b)\right| + \left|P_r^*(b)\right|,$$

$$d^-(K_i) \le d^-(b) + d^*(b) = \left|P_r^-(b)\right| + \left|P_r^*(b)\right|.$$

Одним из критических условий, обеспечивающих полноту трассировки межсоединений и проверяемых как в процессе поиска решения задачи размещения на блочном уровне, так и по ее окончании, является соблюдение ограничений на фактическое количество цепей, пересекающих сечение (разрез) по вертикали для заданного $\boldsymbol{X}$, и фактическое количество цепей, пересекающих сечение (разрез) по горизонтали для заданного $\boldsymbol{Y}$.

С учетом выбранного размещения блоков $\mathcal{P}_\mathrm{b}: \mathrm{K} \to \mathrm{B};\ \mathrm{b_{ij}} = \mathcal{P}_\mathrm{b}(\mathrm{K_k})$ можно вычислить реальные координаты портов блоков (или центров блоков) и, следовательно, граничные координаты цепей пользовательской схемы:

$$X_{min}(n), Y_{min}(n), X_{max}(n), Y_{max}(n), n \in N(s_{mu}).$$

Следовательно, для заданной пользовательской схемы s_{mu} требуемая пропускная способность сечений X_S^k, Y_S^k вычисляется по формулам:

$$\sigma_X\left(s_{mu}, X_S^k\right) = \left|\left\{n:\ n \in N(s_{mu})\ \&\ X_{min}(n) < X_S^k\ \&\ X_{max}(n) \ge X_S^k\right\}\right|,$$

$$\sigma_Y\left(s_{mu}, Y_S^k\right) = \left|\left\{n:\ n \in N(s_{mu})\ \&\ Y_{min}(n) < Y_S^k\ \&\ Y_{max}(n) \ge Y_S^k\right\}\right|.$$

Для обеспечения полной разводимости всего списка проектных цепей необходимо, чтобы требуемое для реализации пользовательской схемы число цепей, пересекающих сечение, превышало фактическую пропускную способность сечения в базовом кристалле не более, чем в p_X и p_Y раз:

$$\sigma_X\left(s_{mu}, X_S^k\right) \le p_X \sigma_X\left(X_S^k\right),$$

$$\sigma_Y\left(s_{mu}, Y_S^k\right) \le p_Y \sigma_Y\left(Y_S^k\right).$$

Предполагается, что коэффициенты p_X и p_Y могут быть подобраны эмпирически для заданного набора пользовательских схем. Выведенное соотношение может быть использовано как во время этапа размещения для оценки трассируемости текущего варианта, так и перед началом процедуры трассировки для определения целесообразности ее выполнения.

5. ЗАКЛЮЧЕНИЕ

В работе представлен теоретико-множественный подход к построению математического аппарата для обобщенного описания пользовательских проектов и базовых кристаллов гетерогенных ПЛИС и реконфигурируемых систем на кристалле. С помощью введенных обозначений формализовано описание коммутационных ресурсов базового кристалла, сформулирована задача трассировки и рассмотрены некоторые аспекты оценки разводимости пользовательских схем. Приведенная математическая модель трассировочных ресурсов позволяет учесть широкий спектр способов программирования коммутационных элементов, а также разнообразие выполняемых ими логических функций. Приведенное формальное описание коммутационных ресурсов базового кристалла и самой процедуры трассировки может быть также использовано в рамках этапа программного прототипирования при анализе пространства архитектурных решений проблемно-ориентированных гетерогенных ПЛИС и реконфигурируемых СнК.

СПИСОК ЛИТЕРАТУРЫ

1. *Тиунов И.В.* Методы ресинтеза схем для ПЛИС на основе ячеек с разделенными выходами и обратной связью // Проблемы разработки перспективных микро- и наноэлектронных систем. 2020 (МЭС-2020). Вып. II. С. 50–56.

2. *Красников Г.Я., Панасенко П.В., Волосов В.А., Щербаков Н.А.* Тенденции развития технологии сложнофункциональной гетерогенной ЭКБ // Международный форум “Микроэлектроника-2018”, 4-я Международная научная конференция “Электронная компонентная база и микроэлектронные модули”. Сборник тезисов. 2018. С. 341–344.

3. *Красников Г.Я.* Возможности микроэлектронных технологий с топологическими размерами менее 5 нм // Наноиндустрия. 2020. Т. 13. № S5-1 (102). С. 13–19.

4. *Красников Г.Я. и др.* Разработка и изготовление на отечественном предприятии по технологии с минимальными топологическими нормами не более 0, 18 мкм библиотеки аналоговых IP блоков для использования в составе сверхбольших интегральных схем “система на кристалле”. Минпромторг РФ, 2017. № 13411.1400099. 11.056.

5. *Эннс В.И.* СнК, БМК или ПЛИС: выбор варианта исполнения цифровой интегральной схемы // Компоненты и технологии. 2018. № 4 (201). С. 100–102.

6. *Hammerquist M., Lysecky R.* Design space exploration for application specific FPGAs in system-on-a-chip designs // 2008 IEEE International SOC Conference, Newport Beach, CA, USA, 2008. P. 279–282.

7. *Parvez H., Marrakchi Z., Kilic A., Mehrez H.* Application-Specific FPGA using heterogeneous logic blocks // ACM Trans. Reconfigurable Technol. Syst., 2011. V. 4. № 3. Article 24. 14 p.

8. *Luu J. et al.* VTR 7.0: Next Generation Architecture and CAD System for FPGAs // ACM Trans. Reconfigurable Technol. Syst., 2014. V. 7. № 2. Article 6. 30 pages.

9. *Nasartschuk K., Herpers R., Kent K.B.* Visualization support for FPGA architecture exploration // 2012 23rd IEEE International Symposium on Rapid System Prototyping (RSP), Tampere, Finland, 2012. P. 128–134.

10. Intel Stratix 10 Logic Array Blocks and Adaptive Logic Modules User Guide / [Электронный ресурс]: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/stratix-10/ug-s10-lab.pdf // Intel, 2020. Дата обращения: 13.04.2021.

11. *Харари Ф.* Теория графов / Пер. с англ. Изд. 5, доп. // М.: Ленанд, 2018. 304 с.

12. *Железников Д.А., Заплетина М.А., Хватов В.М.* Решение задачи трассировки межсоединений для реконфигурируемых систем на кристалле с различными типами коммутационных элементов // Электронная техника. Серия 3: Микроэлектроника, 2018. №. 4. С. 31–36.

13. *McMurchie L., Ebeling C. PathFinder: A* Negotiation-Based Performance-Driven Router for FPGAs // Third International ACM Symposium on Field-Programmable Gate Arrays, Napa Valley, CA, USA, 1995. P. 111–117.

14. *Заплетина М.А., Железников Д.А., Гаврилов С.В.* Иерархический подход к трассировке реконфигурируемой системы на кристалле островного типа // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС), 2020. № 3. С. 16–21.

15. *Жуков Д.В., Железников Д.А., Заплетина М.А.* Применение SAT-подхода к трассировке блоков коммутации для реконфигурируемых систем на кристалле // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС), 2020. № 1. С. 26–32.

16. *Kannan P., Balachandran S., Bhatia D.* On metrics for comparing routability estimation methods for FPGAs // Proceedings 2002 Design Automation Conference (IEEE Cat. No.02CH37324), New Orleans, LA, USA, 2002. P. 70–75.

17. *Fang W.M., Rose J.* Modeling routing demand for early-stage FPGA architecture development // Proceedings of the 16th international ACM/SIGDA symposium on Field programmable gate arrays (FPGA’08). Association for Computing Machinery, New York, NY, USA. P. 139–148.