# Какие архитектуры ПЛИС можно разрабатывать с использованием САПР VTR 8.0

**Андрей СТРОГОНОВ,**
**д. т. н.**
andreistrogonov@mail.ru
**Павел ГОРОДКОВ**
gorodkoff@gmail.com

**В статье предлагается рассмотреть некоторые особенности архитектурного файла САПР VTR 8.0 (Verilog to Routing) [1, 2], используемого для проектирования академических ПЛИС типа FPGA с одноуровневой структурой трассировочных ресурсов, когда конфигурируемые логические блоки (КЛБ) окружены с четырех сторон межсоединениями горизонтальных и вертикальных трассировочных каналов, равномерно распределенных по всей площади кристалла.**

В САПР VTR 8.0 имеются заранее заготовленные различные архитектурные файлы — например, файл *k6_frac_N10_mem32K_40nm.xml* позволяет разработать архитектуру академической ПЛИС, близкую к ПЛИС компании Altera (Intel) серии Stratix IV с умножителями и блоками памяти, а в самой инструкции приводится фрагмент архитектурного файла, описывающий секцию КЛБ ПЛИС компании Xilinx серии Virtex-6 [2].

Будем использовать архитектурный файл ПЛИС, взятый из работы [3], без умножителей и блоков памяти. В работе [3] он приведен не полностью, поэтому мы его восстановили и посмотрели, как с его помощью можно разработать академическую ПЛИС с архитектурой, близкой к ПЛИС серии Stratix III. Предполагается, что ПЛИС будет

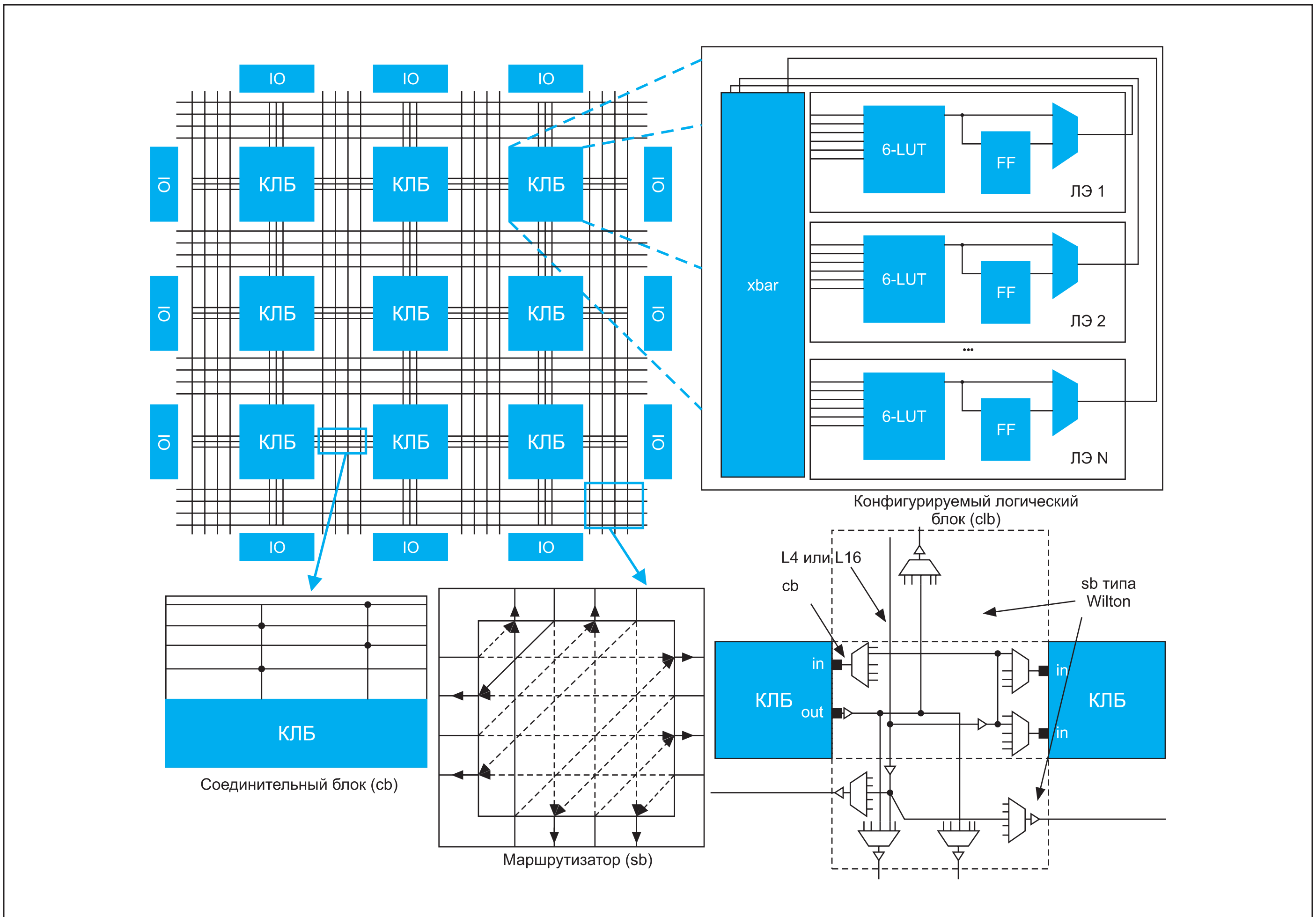


**Рис. 1.** Архитектура академической ПЛИС с одноуровневой структурой межсоединений, используемая в САПР VTR8.0

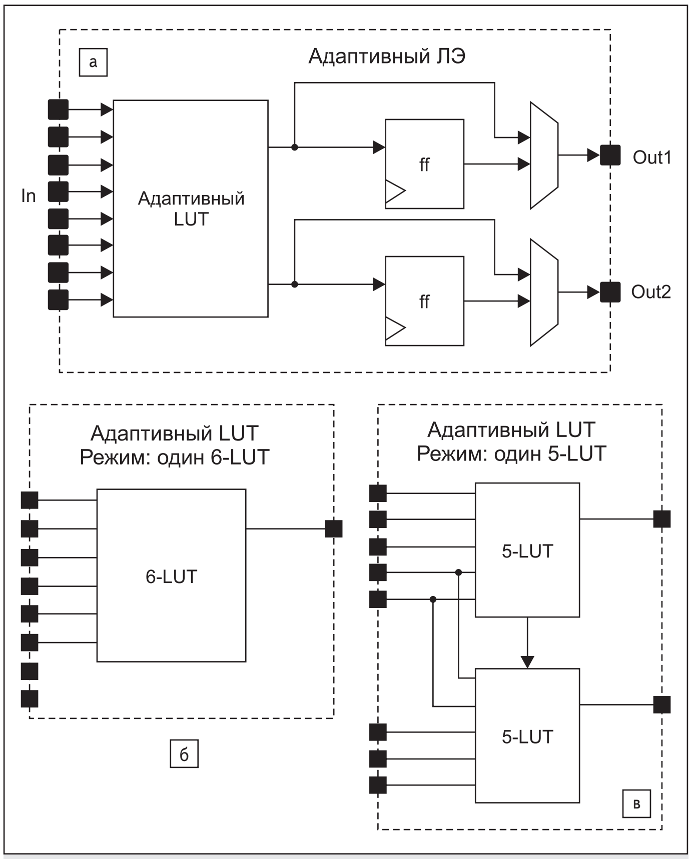


**Рис. 2.** Адаптивный логический элемент и режимы конфигурации LUT:
а) структура адаптивного логического элемента (fle);
б) 6-входовой LUT (ble6);
в) два 5-входовых LUT для организации 8-входового LUT (lut5inter)

изготовлена по 65-нм технологическому КМОП-процессу кремниевой фабрики TSMC с использованием метода стандартных ячеек, применяемого для проектирования заказных цифровых специализируемых БИС [3].

КЛБ (кластер) академической ПЛИС, поддерживаемый САПР VTR 8.0, состоит, в частности, из 10 логических элементов (ЛЭ, или BLE) с числом N = 10 (рис. 1). Логический элемент содержит 6-входовую LUT и триггер [1, 2]. Более подробно с САПР VTR 8.0 можно ознакомиться в работах [4, 5].

Коммутаторы или маршрутизаторы обеспечивают бесконфликтную параллельную передачу информации с множества Y-входов на множество Z-выходов. В зарубежной литературе их называют «кросс-бары», которые бывают полной или частичной (разряженные) коммутации. В данном примере используется маршрутизатор Wilton на мультиплексорных структурах для однонаправленных межсоединений.

Так, в индустриальной ПЛИС серии FLEX 8K компании Altera подключение кластера из восьми ЛЭ с 4-входовыми LUT-таблицами с числом входов 24 к горизонтальному трассировочному каналу из 168 межсоединений осуществляется с помощью мультиплексора частичной коммутации «168 входов × 24 выхода» (разреженный коммутатор на 1/12). Каждая строка (FastTrack, быстрое межсоединение) из канала может быть скоммутирована на входы кластера. Внутри кластера коммутация межсоединений выполняется с помощью мультиплексоров полной коммутации. Это наиболее гибкое устройство, однако его недостатком является большое количество ключей коммутации с соответствующим числом конфигурационной памяти.

Адаптивный логический элемент (fle), предлагаемый в [3], имеет 8-входовую таблицу перекодировок (8-входовая LUTs, или адаптивный LUT) и два триггера. Адаптивный LUT может быть сконфигурирован как чисто 6-входовой LUT (ble6) и/или один триггер или как два 5-входовых с двумя общими входами (lut5inter) и двумя триггерами (рис. 2). LUT КЛБ академической ПЛИС по аналогии с ПЛИС Altera представляет собой LUT адаптивного логического модуля (АЛМ), но только у LUT два режима конфигурации. Например, у ПЛИС Stratix 10 АЛМ состоит из 8-входовой адаптивной LUT, имеющей семь режимов работы, пары сумматоров и четырех триггеров.

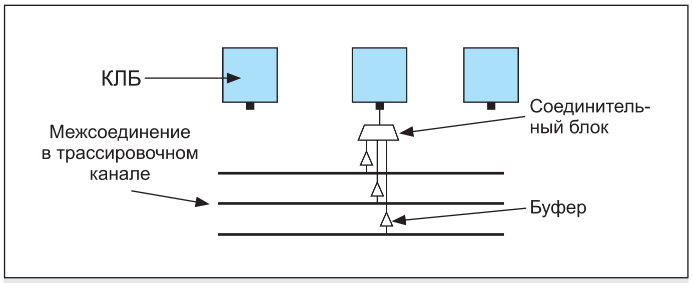


**Рис. 3.** Описание соединительных блоков ПЛИС, подключающих межсоединения из трассировочного канала ко входам кластера

Рассмотрим некоторые особенности формирования архитектурного файла ПЛИС. В теге device задается следующая спецификация: сопротивления *n*-МОП- и *p*-МОП-транзисторов с минимальной шириной канала транзисторов, размер транзисторов в мультиплексах соединительных блоков (ipin); распределение межсоединений в горизонтальном и вертикальном трассировочных каналах, их относительная ширина и тип маршрутизатора в канале. Межсоединения в каналах могут быть как однонаправленные, так и двунаправленные. Однонаправленные межсоединения объединяются в пары.

ipin_mux_trans_size указывается в условных единицах площади, привязанных к минимальной ширине (канала) транзистора для получения минимальной площади. Соединительный блок реализуется как мультиплексор (рис. 3). C_ipin_cblock — входная емкость буфера, изолирующая межсоединение в трассировочном канале от соединительных блоков (мультиплексоров), которые выбирают сигнал для его подключения ко входу КЛБ.

В данном примере задается одинаковая ширина трассировочных *x*- и *y*-каналов ядра кристалла ПЛИС и его периферии и маршрутизатор типа Wilton на мультиплексорных структурах для однонаправленных межсоединений.

```
<device>
  <sizing R_minW_nmos="8926" R_minW_pmos="16067" ipin_mux_trans_size="1.222260"/>
  <timing C_ipin_cblock="1.47e-15" T_ipin_cblock="7.247000e-11"/>
<area grid_logic_tile_area="0"/>
  <chan_width_distr>-
     <x distr="uniform" peak="1.000000"/>
     <y distr="uniform" peak="1.000000"/>
  </chan_width_distr>
  <switch_block type="wilton" fs="3"/>
  </device>
```

Коэффициент объединения по входу fc_in 0,15 означает, что на каждый вход кластера с помощью соединительного блока подключается до 15% доступных межсоединений в трассировочном канале шириной W.

Тег <pinlocations pattern="spread"/> предназначается для связывания контактов КЛБ с трассировочным каналом. Используются специальные шаблоны для указания, с какой стороны располагаются входы/выходы логического блока. Шаблон spread соответствует равномерному распределению контактов со всех сторон КЛБ круговым способом.

Если у КЛБ 33 контакта (входов I = 22, выходов O = 10), то 8 контактов назначаются по всем сторонам, кроме одной, для которой назначены 9 контактов (один из них для подключения синхросигнала).

При использовании двунаправленных межсоединений все маршрутизаторы имеют коэффициент разветвления Fs = 3. То есть, когда

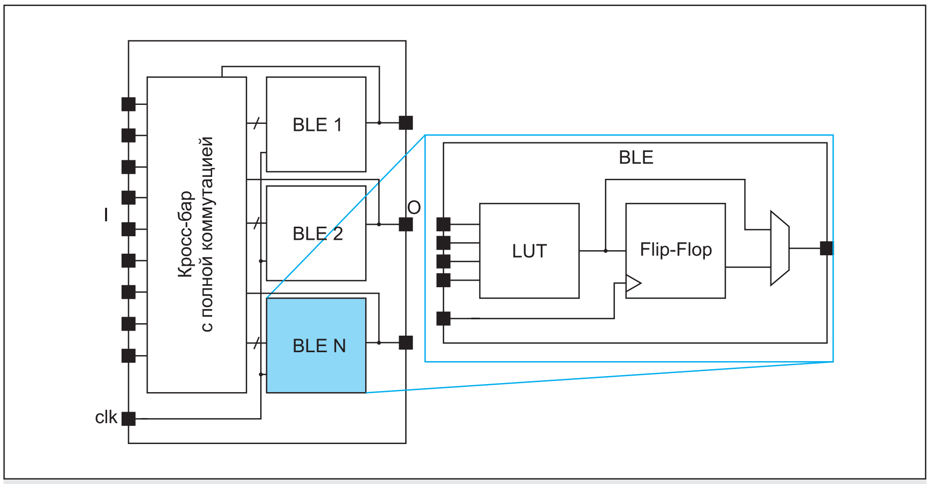


**Рис. 4.** Кросс-бар с полной коммутацией в кластере из N логических элементов академической ПЛИС

горизонтальный и вертикальные трассировочные каналы пересекаются, каждое межсоединение может коммутироваться с тремя другими межсоединениями. Топология маршрутизаторов может быть трех вариантов. Например, планарный или доменный маршрутизатор, используемый в ранних ПЛИС Xilinx серии 4000, в котором двунаправленное межсоединение с порядковым номером 0 может подключаться только к другим межсоединениям с номерами 0 (типа disjoint) и т. д. В маршрутизаторах типа disjoint используются *n*-МОП шеститранзисторные ключи, которые могут быть реализованы с использованием мультиплексоров и буферов с третьим состоянием. Для коммутации однонаправленных межсоединений применяются только мультиплексорные структуры (маршрутизатор Wilton).

Сегодня индустриальные ПЛИС имеют сегментируемую трассировочную структуру с использованием однонаправленных межсоединений (unidirectional).

В общем случае маршрутизатор Wilton является лучшим. В настоящее время считается, что применение однонаправленных межсоединений в совокупности с мультиплексорными структурами в маршрутизаторах наиболее перспективно, так как позволяет получать существенный выигрыш по быстродействию и по площади кристалла, а сам маршрутизатор обладает наилучшей разводимостью.

Содержимое тега <complexblocklist> описывает сложные логические блоки, найденные внутри ПЛИС. Каждый тип сложного логического блока определяется тегом <pb_type name = "string" height = "int"> внутри тега <complexblocklist>. Атрибут name — это имя для сложного блока. Атрибут height указывает, сколько слоев сетки занимает блок.

Внутри сложного блока используется иерархия тегов <pb_type>. Тег верхнего уровня <pb_type> задает сложный блок. Дети <pb_type> — это либо кластеры (которые содержат другие теги <pb_type>), либо примитивы (листья, которые не содержат другие теги <pb_type>).

Кластеры могут включать другие кластеры и примитивы, поэтому нет ограничений на иерархию. Все теги <pb_type> содержат атрибут num_pb = "int", описывающий количество экземпляров конкретного типа кластера или листового блока в этом разделе иерархии. Все теги <pb_type> должны иметь атрибут name = "string", где имя должно быть уникальным в отношении любого родительского, родственного или дочернего тега <pb_type>.

Лист <pb_type> тега может иметь атрибут blif_model = "string". Этот атрибут описывает тип блока в *blif*-файле, поясняя, что этот конкретный лист может реализовать. Например, лист, который реализует LUT, должен иметь blif_model = ".names". Аналогично лист, реализующий .subckt user_block_A, должен иметь атрибут blif_model ="subckt user_block_A". Входные, выходные и/или синхросигналы, порты для этих листов должны соответствовать указанным портам в разделе <models> файла архитектуры.

Описание межсоединений в ПЛИС (тег <interconnect>). Тег mode содержит теги <pb_type> и тег <interconnect>. Ниже описываются теги, используемые в теге <interconnect>.

```
<complete name = "string" input = "string" output = "string" />
```

Обязательные атрибуты:

- name — идентификатор для межсоединения;
- вход — контакты, которые являются входами для этого межсоединения;
- выход — контакты, которые являются выходами этого межсоединения.

На рис. 4 показано использование кросс-бара с полной коммутацией в кластере из ЛЭ (BLE) ранних академических ПЛИС. Наиболее распространенный кластер состоит из N = 10 логических элементов, число входов кластера I = 22, число выходов O = 10, число входов LUT K = 4. Для достижения 98% логического использования кластера требуется число входов, равное 2N+2 (I = 2N+2). Все входы и выходы кластера логически эквивалентны (equivalent="true"). Логическая эквивалентность входов означает, что подключения к этим входам могут быть заменены без изменения их функциональности. Входы в BLE могут поступать либо со входов КЛБ, либо с других входов BLE в логическом блоке через кросс-бар с полной коммутацией.

Кросс-бары в VTR различают с полной коммутацией (complete), с прямой коммутацией (direct) и с шинным мультиплексированием (bus-based multiplexer).

Кросс-бар с полной коммутацией (любой вход коммутатора может быть подключен к любому его выходу) описывается следующей конструкцией (рис. 5):

```
<complete input="Top.in" output="Child.in"/>
```

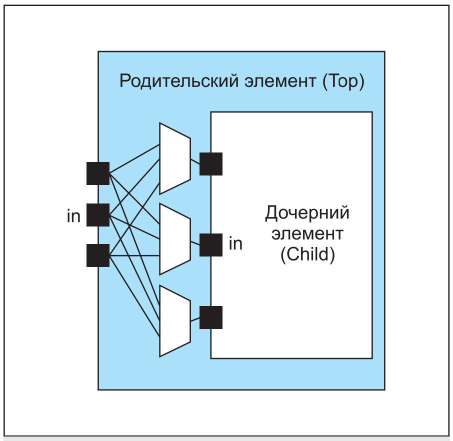


**Рис. 5.** Кросс-бар с полной коммутацией

Прямое подключение входов коммутатора к его выходам по принципу «один в один» (рис. 6):

```
<direct input="Top.in[2:1]" output="Child[1].in"/>
```

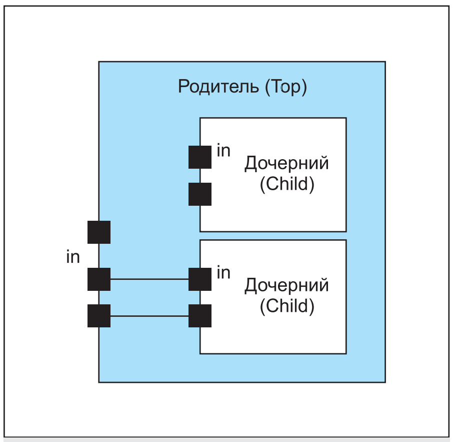


**Рис. 6.** Кросс-бар с прямой коммутацией

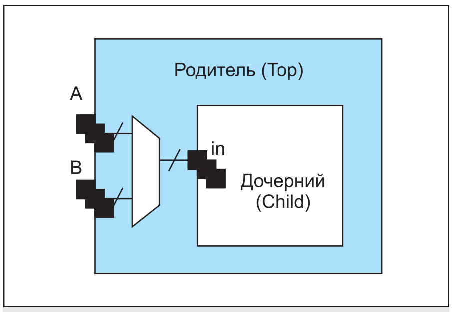


**Рис. 7.** Кросс-бар с шинным мультиплексированием

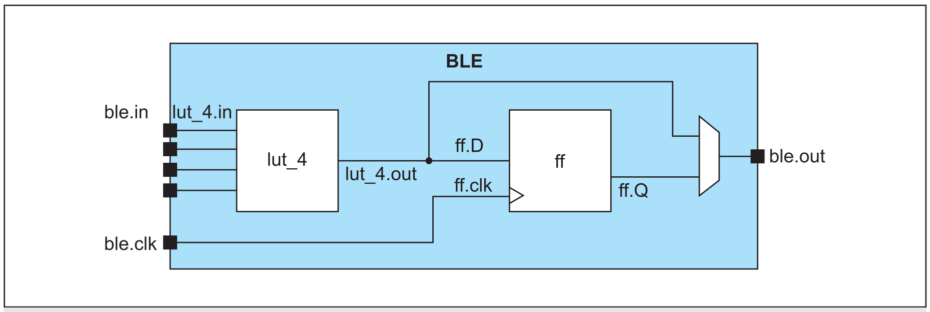


**Рис. 8.** Имена сигналов, используемые для обозначения межсоединений внутри ЛЭ

Кросс-бар с шинным мультиплексированием (рис. 7):

```
<mux input="Top.A Top.B" output="Child.in"/>
```

Кластер (clb) описывается с помощью тега:

```
<pb_type name="clb">
<input name="I" num_pins="22" equivalent="true"/>
<output name="O" num_pins="10" equivalent="true"/>
<clock name="clk" equivalent="false"/>
```

Логический элемент (ble) описывается с помощью тега:

```
<pb_type name="ble" num_pb="10">
<input name="in" num_pins="4"/>
<output name="out" num_pins="1"/>
<clock name="clk"/>
```

В простейшем случае ЛЭ состоит из 4 LUT и триггера, тактируемого фронтом синхросигнала (FF). LUT описывается *blif*-моделью с именем *.names*, а триггер *blif*-моделью с именем *.latch*.

```
<pb_type name="lut_4" blif_model=".names" num_pb="1" class="lut">
<input name="in" num_pins="4" port_class="lut_in"/>
<output name="out" num_pins="1" port_class="lut_out"/>
</pb_type>
<pb_type name="ff" blif_model=".latch" num_pb="1" class="flipflop">
<input name="D" num_pins="1" port_class="D"/>
<output name="Q" num_pins="1" port_class="Q"/>
<clock name="clk" port_class="clock"/>
</pb_type>
```

Ниже представлена коммутация межсоединений ЛЭ с помощью тега <interconnect> (рис. 8):

```
<interconnect>
<direct input="lut_4.out" output="ff.D"/>
<direct input="ble.in" output="lut_4.in"/>
<mux input="ff.Q lut_4.out" output="ble.out"/>
<direct input="ble.clk" output="ff.clk"/>
</interconnect>
</pb_type>
```

Локальная внутрикластерная коммутация, осуществляемая с помощью кросс-баров, также описывается тегом <interconnect>. Посредством мультиплексоров полной коммутации (complete) можно любой вход кластера CLB (clb.I) подключить к любому из входов 10 BLE (BLE[9: 0].in); любой из 10 выходов ЛЭ ble[9:0].out подключить на любой вход кластера clb ble[9:0].in. Синхросигнал clk со входа кластера подключается к тактовым входам триггеров ЛЭ также через мультиплексор полной коммутации:

```
<interconnect>
<complete input="{clb.I ble[9:0].out}" output="ble[9:0].in"/>
<complete input="clb.clk" output="ble[9:0].clk"/>
<direct input="ble[9:0].out" output="clb.O"/>
</interconnect>
```

Тег <complete>, <direct> или <mux> может принимать дополнительный, необязательный тег, называемый <pack_pattern> (шаблон упаковки), который предназначен для описания молекул. Шаблон упаковки представляет собой мощную пользовательскую функцию, указывающую, что инструмент САПР должен группировать определенные атомы netlist (например, LUT, FF, цепи переноса) вместе во время потока САПР.

Это позволяет САПР распознать структуры, имеющие ограниченную гибкость, так что атомы списка соединений, которые соответствуют этим структурам, сохраняются вместе, как единое целое. Данный тег влияет только на работу САПР и не оказывает никакого архитектурного воздействия на определения молекул.

Пример шаблона упаковки:

```
<pack_pattern name="string" in_port="string" out_port="string"/>
```

Обязательные атрибуты:

- name — имя шаблона;
- in_port — входные контакты шаблона;
- out_port — выходные контакты шаблона.

Этот тег дает подсказку инструментам САПР, что некоторые архитектурные структуры должны оставаться вместе во время упаковки. Метка связывает края с именем шаблона. Все примитивы, связанные одним и тем же именем пачки пакетов, становятся единым упаковочным шаблоном. Любая группа атомов в пользовательском списке соединений, соответствующая упаковочному шаблону, будет сгруппирована VPR для образования молекулы. Молекулы хранятся вместе как единое целое в VPR. Это позволяет проектировщику архитектуры ПЛИС помочь инструменту САПР присваивать атомы сложным логическим блокам, которые имеют межсоединения с очень ограниченной гибкостью. Примеры архитектурных структур, для которых можно применять упаковочные шаблоны, включают необязательные зарегистрированные входы/выходы, цепи переноса, блоки с умножением и накоплением и т. д.

В качестве примера рассмотрим классический ЛЭ (BLE), который состоит из LUT и триггера. Если LUT связан с триггером в списке соединений, то проектировщик архитектуры ПЛИС хотел бы, чтобы триггер был упакован с LUT в одном и том же ЛЭ. Чтобы дать VPR указание, что эти блоки должны быть соединены вместе, необходимо обозначить межсоединение, соединяющее LUT и триггер в пару, с помощью шаблона pack_pattern (рис. 9):

```
<pack_pattern name="ble" in_port="lut.out" out_port="ff.D"/>
```

Рассмотрим тег <segmentlist>, описывающий сегментацию межсоединений в трассировочном канале ПЛИС. Содержимое в теге <segmentlist> состоит из группы тегов

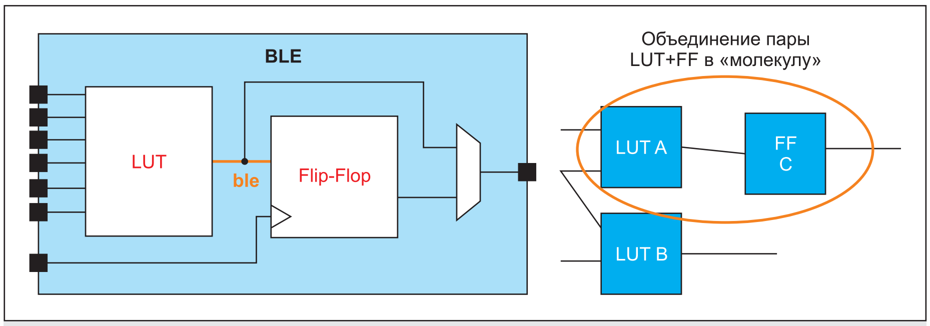


**Рис. 9.** Соединение пары LUT и триггера FF в молекулу с помощью шаблона упаковки с именем ble

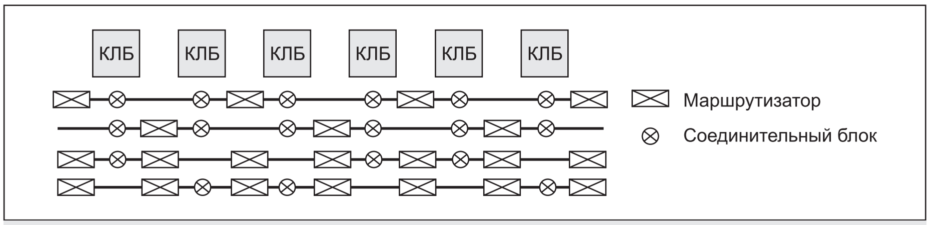


**Рис. 10.** Схемы депопуляции коммутаторов, используемых для реализации маршрутизаторов и соединительных блоков в линиях трассировочного канала

<segment>. Тег <segment> и его содержимое описано ниже.

<segment name = "unique_name" length = "int" type = "{bidir | unidir}" freq = "float" Rmetal = "float".

Обязательные атрибуты:

- name — уникальное буквенно-цифровое имя для идентификации этого типа сегмента;
- length — количество логических блоков (элементов), охватываемых каждым сегментом, или длина сегмента. Ярлык означает, что сегменты этого типа охватывают весь массив FPGA;
- $f_{req}$ — частота появлений межсоединений в трассировочном канале в зависимости от его типа (однонаправленные или двунаправленные межсоединения). VPR автоматически определяет процент межсоединений для каждого типа в зависимости от суммы всех частот. В одном канале может быть несколько межсоединений различной длины, например L4 имеет частоту 0,87, а L16 — 0,13.
- $R_{metal}$ — сопротивление межсоединения на единицу длины (с точки зрения логических блоков), Ом. Например, сегмент длины 5 (охватывает пять блоков) с $R_{metal}$ = 10 Ом/логический блок имеет сквозное сопротивление 50 Ом;
- $C_{metal}$ — емкость межсоединения на единицу длины (с точки зрения логических блоков), Ф. В частности, сегмент длины 5 с $C_{metal}$ = 2e – 14 Ф/логический блок будет иметь общую емкость 10e – 13 Ф.

Ниже показаны примеры депопуляции (удаления) маршрутизаторов (Switch Box) и соединительных блоков (Connection Box) в трассировочном канале с четырьмя линиями (рис. 10).

Так, для длины канала в шесть кластеров (L = 6) для первой линии существует схема депопуляции маршрутизаторов sb 1010101. Вторая линия имеет схему депопуляции 0101010. Единица указывает на существование маршрутизатора в трассировочном канале, а ноль — что в этом месте его нет. В списке целых чисел для L = 6 имеется семь записей. Для значения L должно быть L+1 записей, разделенных пробелами. Для рассмотренных примеров депопуляции маршрутизаторов первая и вторая линии проходят непрерывно через 2 КЛБ.

Соединительный блок описывается следующим тегом.

```
<cb type = "pattern"> int list </ cb>
```

Этот тег описывает депопуляцию соединительных блоков (показано кружками) для конкретного сегмента межсоединения. Для первой линии при L = 6 имеем следующую схему (шаблон) подключения входов/выходов кластера к трассировочному каналу: sb 111111.

Третья линия в трассировочном канале имеет следующий шаблон подключений: 100110. Единица указывает на существование соединительного блока, а ноль указывает, что в этой точке нет соединительного блока. В списке целых чисел при L = 6 должно быть шесть записей, разделенных пробелами.

Восстановленный архитектурный файл академической ПЛИС с архитектурой, близкой к индустриальным ПЛИС серии Stratix III, демонстрирует пример 1. При разработке трассировочной структуры ПЛИС используются межсоединения L4 и L16, схожие с многоканальными соединениями (оптимизированный набор шин различной длины и скорости), применяемыми в архитектуре ПЛИС семейства Altera Stratix II. Так, горизонтальный канал R20 ПЛИС Altera Stratix III является самым быстрым каналом, распространяющимся через 20 кластеров, а R4 обладает наивысшей трассировочной способностью. Также существуют и вертикальные трассировочные каналы C4 и C12.

Рассмотрим 2-разрядный сумматор, реализованный в академической ПЛИС, по своей архитектуре схожей с архитектурой ПЛИС серии Straix III. Предварительно *blif*-файл сумматора подвергнут оптимизации программным инструментом ABC.

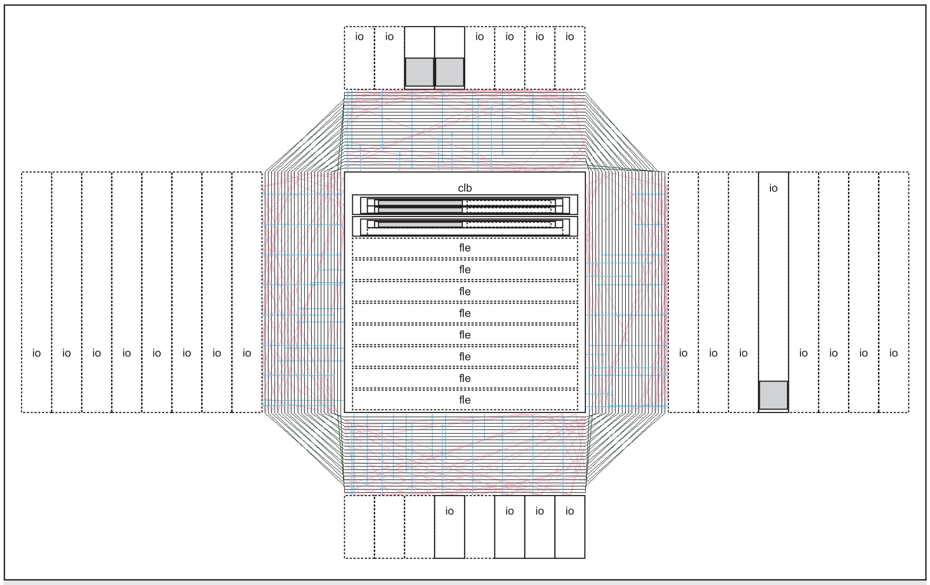


**Рис. 11.** Двухразрядный сумматор в ПЛИС

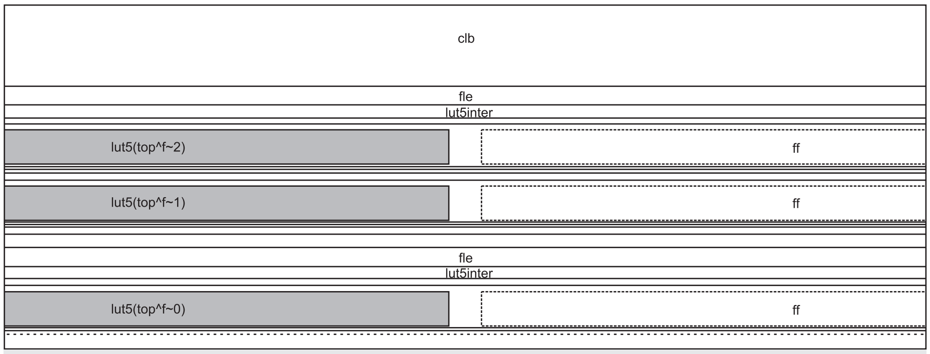


**Рис. 12.** Для реализации сумматора требуется два блока lut5inter в режиме n2_lut5 (задействованные lut5 подсвечены серым цветом)

```
<architecture>
<models>
</models>
<layout>
  <auto_layout aspect_ratio="1.0">
     <!--Perimeter of 'io' blocks with 'EMPTY' blocks at corners-->
     <perimeter type="io" priority="100"/>
     <corners type="EMPTY" priority="101"/>
     <!--Fill with 'clb'-->
     <fill type="clb" priority="1"/>
     </auto_layout>
</layout>
<device>
  <sizing R_minW_nmos="8926" R_minW_pmos="16067" ipin_mux_trans_size="1.222260"/>
  <timing C_ipin_cblock="1.47e-15" T_ipin_cblock="7.247000e-11"/>
  <area grid_logic_tile_area="0"/>
  <chan_width_distr>
     <x distr="uniform" peak="1.000000"/>
     <y distr="uniform" peak="1.000000"/>
  </chan_width_distr>
  <switch_block type="wilton" fs="3"/>
  </device>
<switchlist>
  <switch type="mux" name="0" R="551" Cin=".77e-15" Cout="4e-15" Tdel="58e-12" mux_trans_size="2.630740" buf_size="27.645901"/>
</switchlist>
<segmentlist>
&lt;segment freq="0.87" length="4" type="unidir" Rmetal="101" Cmetal="22.5e-15">
<mux name="0"/>
<sb type="pattern">1 1 1 1 1 </sb>
<cb type="pattern">1 1 1 1 </cb>
&lt;/segment>
&lt;segment freq="0.13" length="16" type="unidir" Rmetal="101" Cmetal="22.5e-15">
<mux name="0"/>
<sb type="pattern">1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 </sb>
<cb type="pattern">1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</cb>
&lt;/segment>
</segmentlist>
<complexblocklist>
<pb_type name="io" capacity="8" area="0">
     <input name="outpad" num_pins="1"/>
     <output name="inpad" num_pins="1"/>
     <clock name="clock" num_pins="1"/>
     <mode name="inpad">
     <pb_type name="inpad" blif_model=".input" num_pb="1">
        <output name="inpad" num_pins="1"/>
     </pb_type>
     <interconnect>
        <direct name="inpad" input="inpad.inpad" output="io.inpad">
        <delay_constant max="4.243e-11" in_port="inpad.inpad" out_port="io.inpad"/>
        </direct>
     </interconnect>
     </mode>
     <mode name="outpad">
     <pb_type name="outpad" blif_model=".output" num_pb="1">
        <input name="outpad" num_pins="1"/>
     </pb_type>
     <interconnect>
        <direct name="outpad" input="io.outpad" output="outpad.outpad">
        <delay_constant max="1.394e-11" in_port="io.outpad" out_port="outpad.outpad"/>
        </direct>
     </interconnect>
     </mode>
     <fc default_in_type="frac" default_in_val="0.15" default_out_type="frac" default_out_val="0.10"/>
     <pinlocations pattern="custom">
     <loc side="left">io.outpad io.inpad io.clock</loc>
     <loc side="top">io.outpad io.inpad io.clock</loc>
     <loc side="right">io.outpad io.inpad io.clock</loc>
     <loc side="bottom">io.outpad io.inpad io.clock</loc>
     </pinlocations>
     <!-- Place I/Os on the sides of the FPGA -->
     <power method="ignore"/>
  </pb_type>
  <!-- Define I/O pads ends -->

<pb_type name="clb" area="53894">
<input name="I1" num_pins="13" equivalent="true"/>
<input name="I2" num_pins="13" equivalent="true"/>
<input name="I3" num_pins="13" equivalent="true"/>
<input name="I4" num_pins="13" equivalent="true"/>
<output name="O" num_pins="20" equivalent="true"/>
<clock name="clk" num_pins="1"/>

<pb_type name="fle" num_pb="10">
<input name="in" num_pins="8"/>
<output name="out" num_pins="2"/>
<clock name="clk" num_pins="1"/>
<!-- Dual 5-LUT mode definition begin -->
<mode name="n2_lut5">
<pb_type name="lut5inter" num_pb="1">
<input name="in" num_pins="8"/>
<output name="out" num_pins="2"/>
<clock name="clk" num_pins="1"/>

<pb_type name="ble5" num_pb="2">
<input name="in" num_pins="5"/>
<output name="out" num_pins="1"/>
<clock name="clk" num_pins="1"/>

<mode name="blut5">
<pb_type name="flut5" num_pb="1">
<input name="in" num_pins="5"/>
<output name="out" num_pins="1"/>
<clock name="clk" num_pins="1"/>
<pb_type name="lut5" blif_model=".names" num_pb="1" class="lut">
<input name="in" num_pins="5" port_class="lut_in"/>
<output name="out" num_pins="1" port_class="lut_out"/>
<!-- LUT timing using delay matrix -->
  <delay_matrix type="max" in_port="lut5.in" out_port="lut5.out">
        261e-12
        261e-12
        261e-12
        261e-12
        261e-12
        </delay_matrix>
</pb_type>
  <pb_type name="ff" blif_model=".latch" num_pb="1" class="flipflop">
  <input name="D" num_pins="1" port_class="D"/>
  <output name="Q" num_pins="1" port_class="Q"/>
  <clock name="clk" num_pins="1" port_class="clock"/>
  <T_setup value="66e-12" port="ff.D" clock="clk"/>
  <T_clock_to_Q max="124e-12" port="ff.Q" clock="clk"/>
  </pb_type>
<interconnect>
  <direct name="direct1" input="flut5.in" output="lut5.in"/>
  <direct name="direct2" input="lut5.out" output="ff.D">
  <pack_pattern name="ble5" in_port="lut5.out" out_port="ff.D"/>
  </direct>
<direct name="direct3" input="flut5.clk" output="ff.clk"/>
<mux name="mux1" input="ff.Q lut5.out" output="flut5.out">
<delay_constant max="25e-12" in_port="lut5.out" out_port="flut5.out"/>
<delay_constant max="45e-12" in_port="ff.Q" out_port="flut5.out"/>
</mux>
  </interconnect>
</pb_type>
<interconnect>
  <direct name="direct1" input="ble5.in" output="flut5.in"/>
  <direct name="direct2" input="ble5.clk" output="flut5.clk"/>
  <direct name="direct3" input="flut5.out" output="ble5.out"/>
</interconnect>
</mode>
</pb_type>
<interconnect>
<direct name="direct1" input="lut5inter.in[6]" output="ble5[0:0].in[4]"/>
<direct name="direct2" input="lut5inter.in[3:0]" output="ble5[0:0].in[3:0]"/>
<direct name="direct4" input="lut5inter.in[7]" output="ble5[1:1].in[4]"/>
<direct name="direct5" input="lut5inter.in[5:4]" output="ble5[1:1].in[3:2]"/>
<direct name="direct6" input="lut5inter.in[1:0]" output="ble5[1:1].in[1:0]"/>
<direct name="direct3" input="ble5[1:0].out" output="lut5inter.out"/>
<complete name="complete1" input="lut5inter.clk" output="ble5[1:0].clk"/>
</interconnect>
</pb_type>
<interconnect>
  <direct name="direct1" input="fle.in" output="lut5inter.in"/>
  <direct name="direct2" input="lut5inter.out" output="fle.out"/>
  <direct name="direct3" input="fle.clk" output="lut5inter.clk"/>
</interconnect>
</mode>

<!-- 6-LUT mode definition begin -->
<mode name="n1_lut6">
<!-- Define 6-LUT mode -->

<pb_type name="ble6" num_pb="1">
<input name="in" num_pins="6"/>
<output name="out" num_pins="1"/>
<clock name="clk" num_pins="1"/>
<pb_type name="lut6" blif_model=".names" num_pb="1" class="lut">
<input name="in" num_pins="6" port_class="lut_in"/>
<output name="out" num_pins="1" port_class="lut_out"/>
<!-- LUT timing using delay matrix -->
<delay_matrix type="max" in_port="lut6.in" out_port="lut6.out">
        261e-12
        261e-12
        261e-12
        261e-12
        261e-12
        261e-12
        </delay_matrix>
</pb_type>
  <pb_type name="ff" blif_model=".latch" num_pb="1" class="flipflop">
  <input name="D" num_pins="1" port_class="D"/>
  <output name="Q" num_pins="1" port_class="Q"/>
  <clock name="clk" num_pins="1" port_class="clock"/>
  <T_setup value="66e-12" port="ff.D" clock="clk"/>
  <T_clock_to_Q max="124e-12" port="ff.Q" clock="clk"/>
  </pb_type>

<interconnect>
  <direct name="direct1" input="ble6.in" output="lut6[0:0].in"/>
  <direct name="direct2" input="lut6.out" output="ff.D">
  <pack_pattern name="ble6" in_port="lut6.out" out_port="ff.D"/>
  </direct>
<direct name="direct3" input="ble6.clk" output="ff.clk"/>
<mux name="mux1" input="ff.Q lut6.out" output="ble6.out">
<delay_constant max="25e-12" in_port="lut6.out" out_port="ble6.out"/>
<delay_constant max="45e-12" in_port="ff.Q" out_port="ble6.out"/>
</mux>
  </interconnect>
</pb_type>
<interconnect>
  <direct name="direct11" input="fle.in[3:0]" output="ble6.in[3:0]"/>
  <direct name="direct12" input="fle.in[7:6]" output="ble6.in[5:4]"/>
<direct name="direct2" input="ble6.out" output="fle.out[0:0]"/>
  <direct name="direct3" input="fle.clk" output="ble6.clk"/>
</interconnect>
</mode>
</pb_type>
<interconnect>
<complete name="lutA" input="clb.I4 clb.I3 fle[1:0].out fle[3:2].out fle[8:8].out"
output="fle[9:0].in[0:0]">
<!-- delay matrix -->
</complete>
<complete name="lutB" input="clb.I3 clb.I2 fle[3:2].out fle[5:4].out fle[9:9].out"
output="fle[9:0].in[1:1]">
<!-- delay matrix -->
</complete>
<complete name="lutC" input="clb.I2 clb.I1 fle[5:4].out fle[7:6].out fle[8:8].out"
output="fle[9:0].in[2:2]">
<!-- delay matrix -->
</complete>
<complete name="lutD" input="clb.I4 clb.I2 fle[1:0].out fle[5:4].out fle[9:9].out"
output="fle[9:0].in[3:3]">
<!-- delay matrix -->
</complete>
<complete name="lutE" input="clb.I3 clb.I1 fle[3:2].out fle[7:6].out fle[8:8].out"
output="fle[9:0].in[4:4]">
<!-- delay matrix -->
</complete>
<complete name="lutF" input="clb.I4 clb.I1 fle[1:0].out fle[7:6].out fle[9:9].out"
output="fle[9:0].in[5:5]">
<!-- delay matrix -->
</complete>
<complete name="lutG" input="clb.I4 clb.I3 fle[1:0].out fle[3:2].out fle[8:8].out"
output="fle[9:0].in[6:6]">
<!-- delay matrix -->
</complete>
<complete name="lutH" input="clb.I3 clb.I2 fle[3:2].out fle[5:4].out fle[9:9].out"
output="fle[9:0].in[7:7]">
<!-- delay matrix -->
</complete>
<complete name="clks" input="clb.clk" output="fle[9:0].clk">
</complete>
<direct name="clbouts1" input="fle[9:0].out[0:0]" output="clb.O[9:0]"/>
<direct name="clbouts2" input="fle[9:0].out[1:1]" output="clb.O[19:10]"/>
</interconnect>

<fc default_in_type="frac" default_in_val="0.055"
default_out_type="frac" default_out_val="0.10"/>
<pinlocations pattern="spread"/>

</pb_type>
</complexblocklist>
</architecture>
```

**Пример 1.** Восстановленный архитектурный файл из работы [3]

На рис. 11 видно, что межсоединения L4 и L16 не используются из-за простоты проекта, показан двухразрядный сумматор, реализованный в ПЛИС. Применяется только периферийный трассировочный канал для подключения элементов ввода/вывода ко входам КЛБ. Проект занимает всего 1 clb. Для реализации сумматора требуются два адаптивных логических элемента fle типа lut5inter в режиме n2_lut5 (рис. 12) или три lut5. В одном lut5inter задействуются два lut5, а в другом один lut5 (задействованные lut5 подсвечены серым цветом).

Из тестов производительности САПР VTR 7.0 (Verilog-проекты) возьмем файл r*aygentop.v* и предварительно подвергнем оптимизации с помощью ABC. Для реализации проекта требуется 1029 КЛБ (рис. 13). Ширина трассировочного канала 150 однонаправленных межсоединений. В более сложном проекте уже предусмотрены адаптивные логические элементы fle двух типов: ble6 и lut5inter (рис. 14). Детальное раскрытие структуры fle типов ble6 и lut5inter демонстрирует рис. 15. На рис. 16 подсвечены синим цветом межсоединения L4 (по вертикали) и L16 (по горизонтали), проходящие непрерывно через 4 КЛБ и 16 КЛБ.

На рис. 17 детально раскрыта структура маршрутизатора Wilton в трассировочном канале. Треугольники (серые и черные) по периферии маршрутизатора показывают направление передачи сигналов по межсоединениям, которые проходят через маршрутизатор непрерывно. На концах однонаправленных межсоединений L4 (направление распространения сверху вниз), которые сегментируются маршрутизатором, находятся мультиплексоры «12 в 1» (подсвечены желтым). На рис. 18 показаны маршрутизаторы Wilton в трассировочном канале (фрагменты). Синим цветом выделены соединительные блоки, красным — выходы КЛБ, которые заводятся непосредственно в маршрутизаторы (отображены зеленым цветом). Синим цветом также отмечены входы, а красным — выходы КЛБ.

## Выводы

САПР VTR 8.0 позволяет создавать и исследовать новые архитектуры академических ПЛИС, по своим техническим характеристикам близкие к индустриальным ПЛИС с адаптивными логическими элементами.

## Литература


1. Luu J., Goeders J., Wainberg M., Somerville A., Yu T., Nasartschuk K., Nasr M., Wang S., Liu T., Ahmed N., Kent K. B., Anderson J., Rose J., Betz V. VTR 7.0: Next Generation Architecture and CAD System for FPGAs // ACM Transactions on Reconfigurable Technology and Systems. 2014. Vol. 7. No. 2. Art. 6.


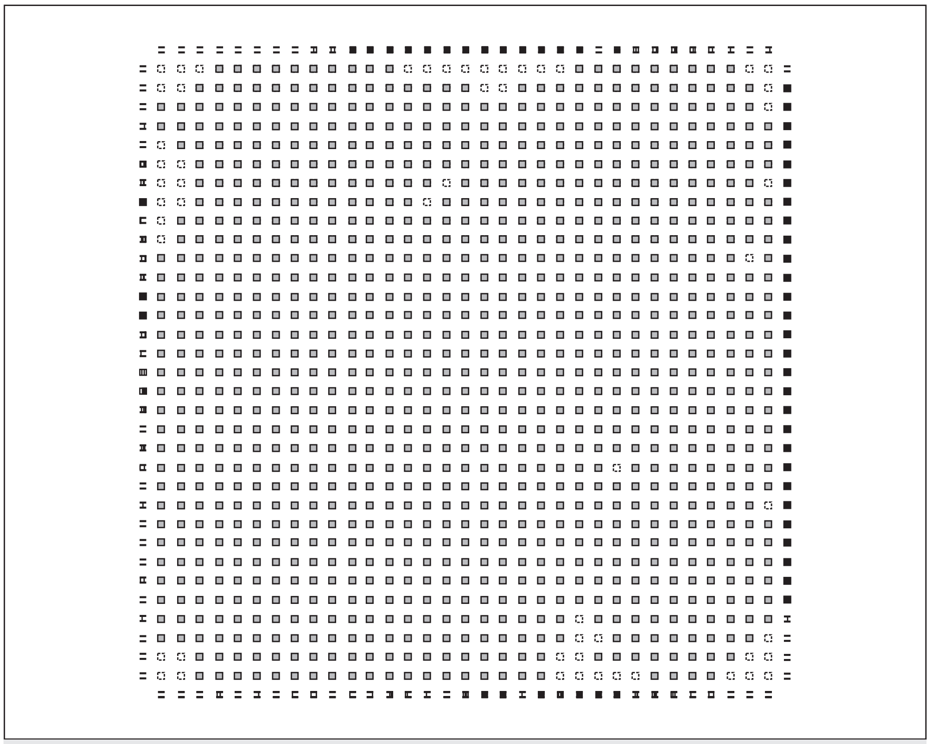

**Рис. 13.** Реализация проекта raygentop.v в академческую ПЛИС

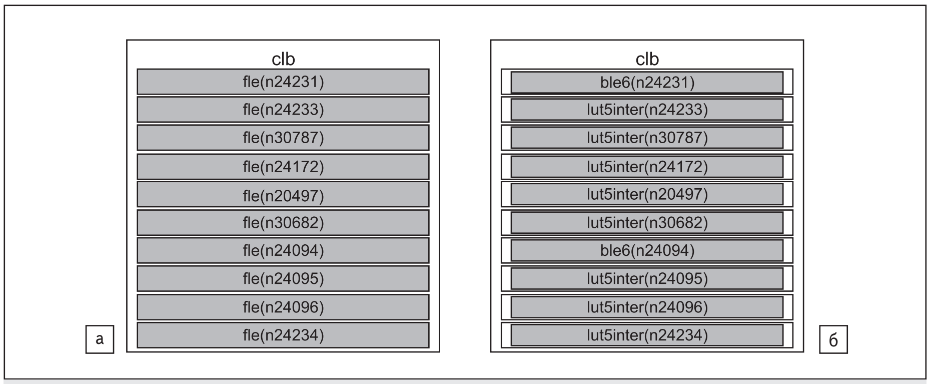


**Рис. 14.** а) Кластер (clb) из 10 адаптивных fle;
б) использование адаптивных логических элементов fle двух типов: ble6 и lut5inter

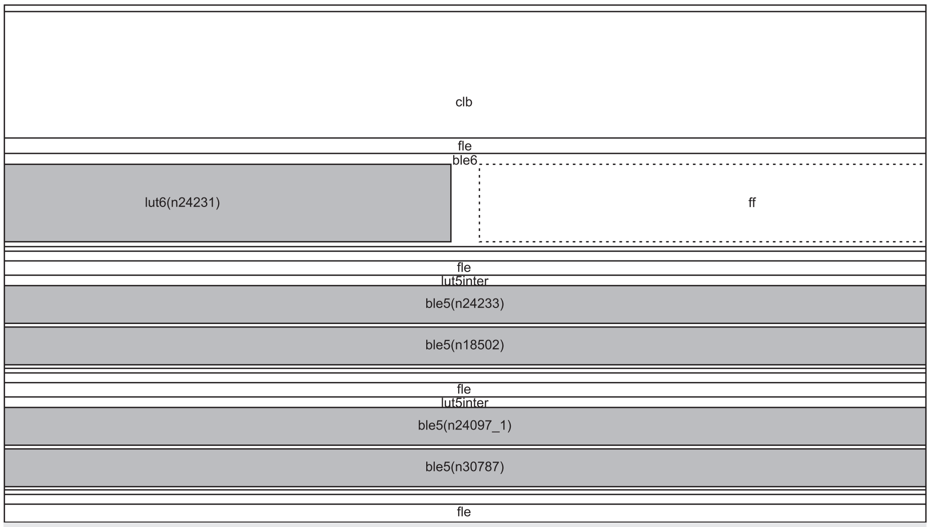


**Рис. 15.** Детальное раскрытие структуры fle двух типов: ble6 и lut5inter

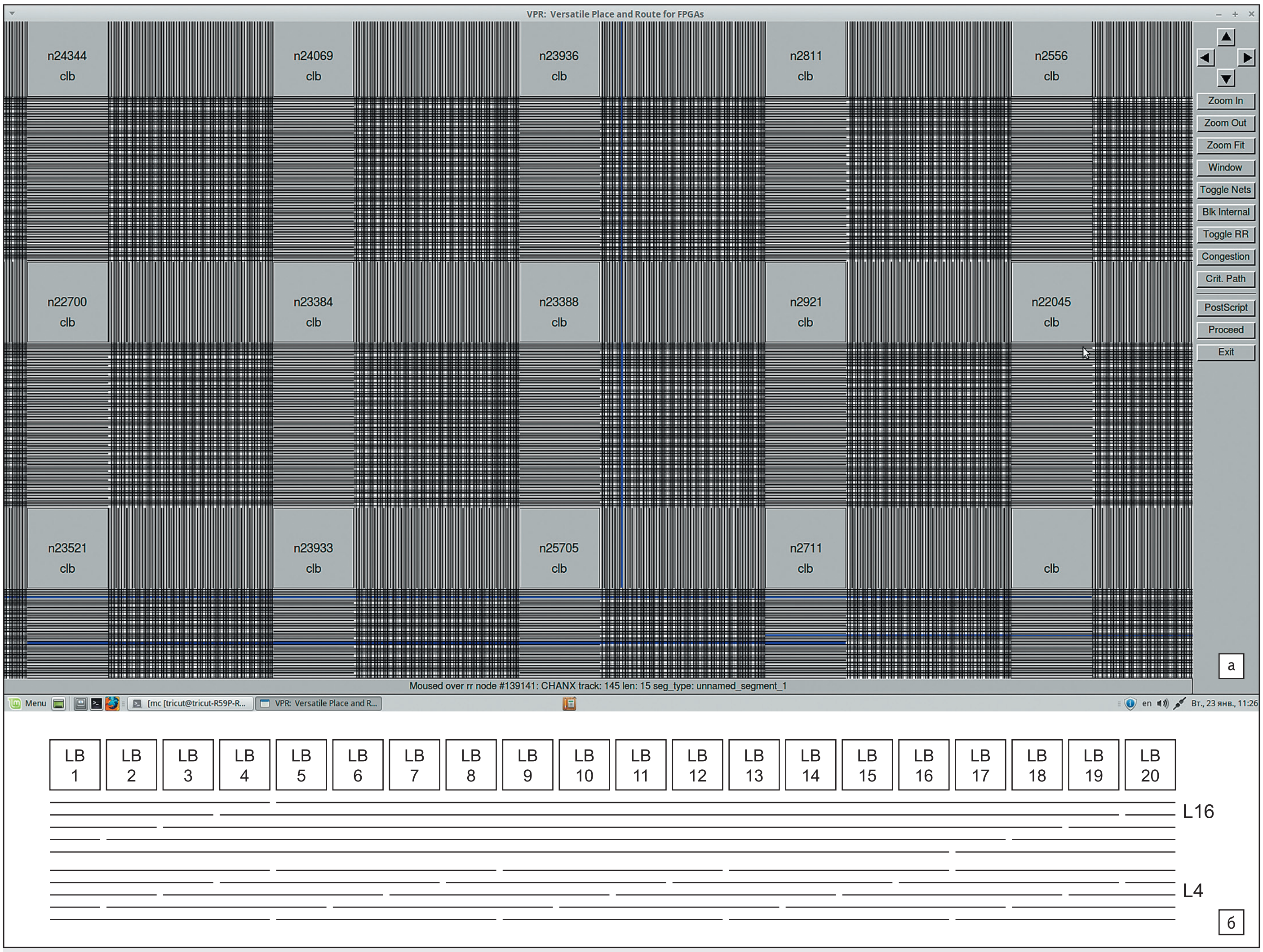


**Рис. 16.** а) Выделены синим цветом межсоединения L4 (по вертикали) и L16 (по горизонтали), проходящие непрерывно через 4 КЛБ и 16 КЛБ;
б) упрощенное представление межсоединений L16 и L4

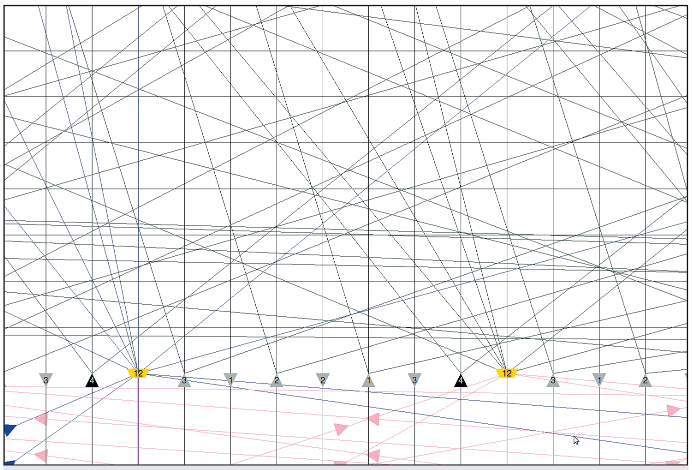

**Рис. 17.** Детальное раскрытие структуры маршрутизатора Wilton в трассировочном канале. Мультиплексоры «12 в 1» отмечены желтым цветом

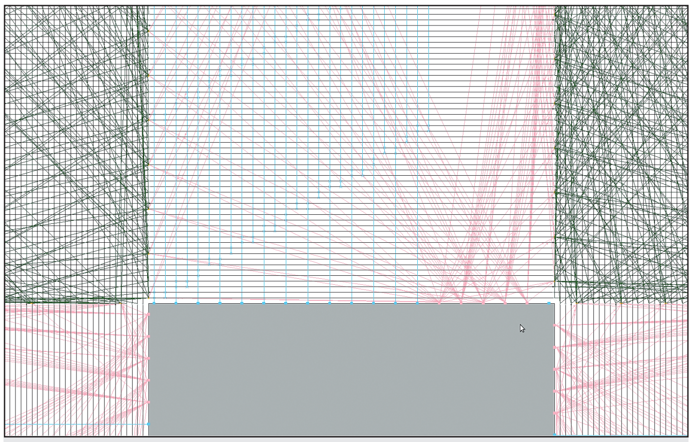

**Рис. 18.** Маршрутизаторы Wilton в трассировочном канале (фрагменты). Синим цветом указаны соединительные блоки, красным — выходы КЛБ, которые заводятся непосредственно в маршрутизаторы (выделены зеленым цветом)


2. Verilog-to-Routing Documentation. Release 7.0.7. Aug 23, 2017.
3. Kim J. H., Anderson J. H. Synthesizable Standard Cell FPGA Fabrics Targetableby the Verilog-to-Routing CAD Flow // ACM Transactions on Recon-figurable Technology and Systems. 2017. Vol. 10. No. 2. Art. 11. www.dx.doi.org/10.1145/3024063
4. Строгонов А., Цыбин С., Городков П. САПР VTR7 для проектирования академических ПЛИС // Компоненты и технологии. 2016. № 3.
5. Строгонов А., Городков П. Реализация Verilog-проектов в базисе академических ПЛИС с применением САПР VTR 7.0 // Компоненты и технологии. 2017. № 5.