Проектирование для ПЛИС Xilinx: системные аспекты и уровень регистровых передач

Илья ТАРАСОВ,
д. т. н.
ilya_e_tarasov@mail.ru

В статье рассматриваются вопросы проектирования для ПЛИС Xilinx, требующие выяснения на ранних этапах данного процесса. Несмотря на универсальные возможности программируемых логических ячеек и постоянное совершенствование FPGA, остается вероятность заложить в проект архитектурные решения, неоптимальные для ПЛИС как таковых или для FPGA компании Xilinx в частности. Тем не менее существует ряд достаточно универсальных рекомендаций, которые могут быть успешно применены для широкого круга проектов на базе ПЛИС.

Введение

Часто на ранних этапах разработки проекта следует заранее обратить внимание на ряд технических вопросов, поскольку в противном случае специалист может столкнуться с ситуацией, когда достигнутые показатели будут неудовлетворительными. Для повышения качества проекта придется провести его глубокий пересмотр, а в худшем варианте — переделать печатную плату, изменив расстановку выводов ПЛИС и добавив новые компоненты и сигналы. Иногда гибкость и перепрограммируемость FPGA формирует неверное представление о том, что все вопросы можно разрешить впоследствии, а пока достаточно просто соединить все внешние сигналы с выводами ПЛИС почти в произвольном порядке. Однако для больших кристаллов и при необходимости достижения высоких частот и высокой плотности упаковки проекта такой подход оказывается неудачным.

Кроме того, при описании схем на соответствующих языках нужно придерживаться рекомендаций Xilinx, чтобы итоговый результат был хорошо адаптирован к архитектуре ПЛИС и действительным возможностям элементной базы. В конечном итоге оказывается, что на ранних этапах этого процесса можно заранее выполнить целый ряд технических мероприятий, которые существенно облегчат разработку, отладку и сопровождение проекта.

Выбор инструментов описания проекта

В настоящее время кроме основной САПР Vivado, поддерживающей полный цикл проектирования от ввода исходных текстов до программирования ПЛИС, возможно использование вспомогательных инструментов проектирования, ориентированных на решение отдельных задач. Эти инструменты име-

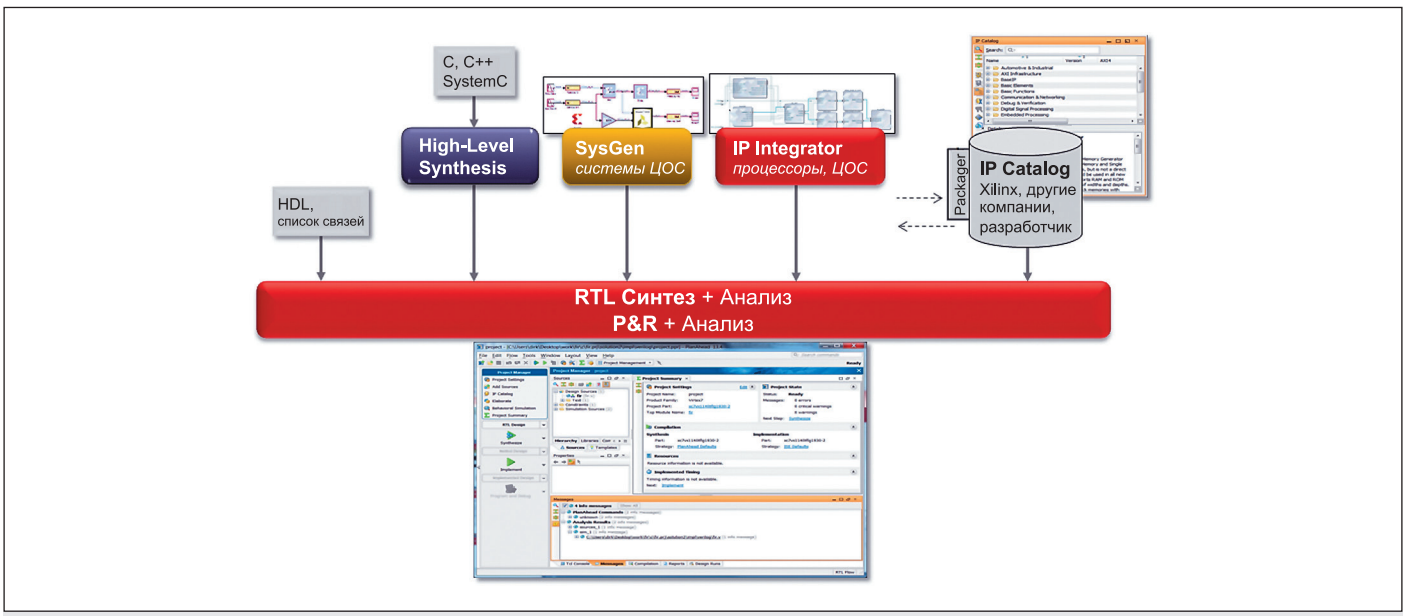

Рис. 1. Инструменты описания проектов для ПЛИС Xilinx

ют как преимущества, так и недостатки, и очевидно, что их применение для неподходящих целей приведет как минимум к снижению качества получаемых результатов.

Основным способом описания проекта является ввод исходных текстов на языках описания аппаратуры — VHDL и Verilog. В новых версиях Vivado к ним добавлен System Verilog, который расширяет возможности Verilog в области моделирования и верификации, однако вряд ли его надо считать новым классом средств проектирования (рис. 1).

Дополнительные инструменты проектирования имеют преимущественные области применения, в которых они могут показать наилучшие результаты.

1. High-Level Synthesis (HLS) предполагает ввод описаний на языках C/C++/SystemC. Это не означает, что произвольный алгоритм на указанных языках, предназначенный для персонального компьютера, можно напрямую перенести в FPGA. Главное преимущество заключается в том, что разработчик освобождается от необходимости управлять последовательностью операций, автоматически поддерживаются многие типы данных (в том числе с плавающей точкой). Алгоритмы получат существенное ускорение при наличии в исходной задаче операций, выполняющихся параллельно. Разработанный на HLS модуль может быть легко подключен к проекту в System Generator или к процессорной системе в качестве периферийного модуля с интерфейсом AXI4.

В то же время результаты работы HLS имеют две особенности, способные затруднить применение созданных модулей:

- сильная зависимость результатов от директив компилятора и стиля кодирования;
- интервальный характер латентности, плохо предсказуемое потактовое поведение синтезированной схемы.

В целом HLS не очень хорошо подходит для компактных проектов с малой латентностью, основанных на хорошо известных разработчику схемотехнических решениях. Например, простой КИХ-фильтр имеет несложное RTL-представление и может быть реализован на VHDL или Verilog, или в виде IP-ядра, а HLS представляется несколько избыточным. Кроме того, при реализации КИХ-фильтра необходимо строго следить за постоянством временного интервала между обработкой отдельных отсчетов, и переменный характер латентности здесь недопустим. Напротив, если речь идет о проекте, ориентированном на длительное моделирование, выполнение различных экспериментов, то высокая скорость разработки на HLS является значимым преимуществом.

2. System Generator for DSP (Sysgen) обеспечивает возможность моделировать системы цифровой обработки сигналов в связке инструментов Matlab + Simulink.

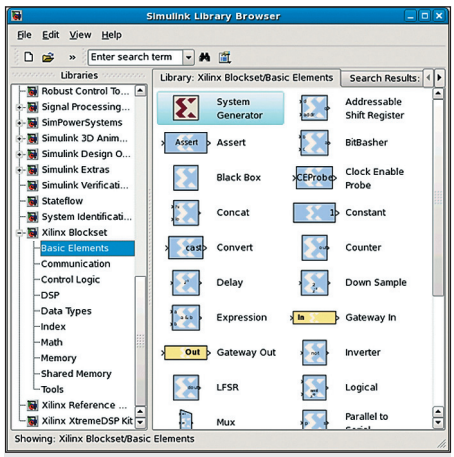

Рис. 2. Вкладка Xilinx Blockset в библиотеке Simulink

На рис. 2 показана палитра компонентов Xilinx Blockset, которая добавляется в Simulink. Все компоненты ссылаются на IP-ядра Xilinx, поэтому характеристики такой схемы будут предсказуемыми, а частота — достаточно высокой.

При моделировании в Simulink можно пользоваться всеми возможностями, предоставляемыми этой САПР в части создания генераторов, систем преобразования данных, наблюдения и связи со сторонними программными продуктами. Это открывает широкие возможности для предварительного системного моделирования устройств DSP. После того как результаты моделирования будут признаны удовлетворительными, полученную схему можно экспортировать в проект в виде IP-ядер и/или RTL-представления (списка связей).

Достоинством Sysgen является высокая продуктивность создания систем цифровой обработки сигналов и хорошая предсказуемость характеристик производительности и занимаемых ресурсов ПЛИС. Недостатком — слабая пригодность этого инструмента для решения посторонних задач, не относящихся к цифровой обработке сигналов, поскольку возможности Sysgen ограничиваются теми компонентами, которые имеются в готовых библиотеках.

3. Embedded Development Kit (EDK) служит для быстрой настройки встраиваемой процессорной подсистемы на базе софт-процессора MicroBlaze (все семейства ПЛИС) или ARM Cortex-A (только семейство Zynq с аппаратными ядрами ARM). Процессоры поддержаны компилятором gcc, и в состав EDK входит IDE Eclipse, предназначенная для разработки и отладки программного обеспечения для этих процессоров. Визуальный инструмент Xilinx Processor Stiduo позволяет быстро сконфигурировать процессор и его периферийные устройства, как выбираемые из стандартных библиотек, так и выполняемые отдельно. Процессорные ядра используют популярную для платформы ARM системную шину AXI4, и практически все инструменты Xilinx позволяют достаточно просто подключить созданный компонент к этой шине.

Преимуществом применения процессора следует назвать возможность привнесения в проект всех инструментов из мира программирования. Так, использование столь распространенного компилятора, как gcc (включая поддержку работы ОС Linux на ПЛИС), и широкого набора библиотек помогает быстро добавить привычный набор периферийных устройств и интерфейсов. В качестве примера стоит упомянуть поддержку памяти DDR2/DDR3, USB, Ethernet, в том числе стек протоколов TCP/IP, PCI Express, и т. д. В данном случае может оказаться очень важным, что такой подход позволяет перенести значительную часть процесса отладки из аппаратной в программную часть. Если разработчики не вполне уверены в корректности используемых ими алгоритмов, то для их аппаратной реализации придется регулярно перезапускать процесс получения конфигурации ПЛИС. Каждая итерация данного процесса выполняется гораздо дольше, чем компиляция сопоставимой по функциональности программы. Вот почему замена отладки аппаратной части на отладку программного обеспечения встроенного процессора резко ускоряет процесс разработки.

Нельзя не отметить и крайне отличающуюся сложность аппаратной и программной реализации некоторых задач. Например, полный стек протоколов TCP/IP практически невозможно реализовать в ПЛИС, тогда как для gcc имеется несколько реализаций, которые могут быть скомпилированы для ARM/MicroBlaze.

Таким образом, явным преимуществом использования EDK становится возможность быстрого добавления в проект процессорной подсистемы и получение доступа ко всей полноте соответствующих программных инструментов. Недостатком, подобно Sysgen, является строгая специализация данного инструмента на конфигурировании процессорных систем. Дополнительные периферийные устройства могут быть созданы на базе RTL-описаний, HLS или Sysgen.

Совместное проектирование печатной платы и ПЛИС

Несмотря на то, что пользовательские выводы ПЛИС в целом полагаются взаимозаменяемыми, существует целый ряд причин, по которым следует изучить вопрос распределения внешних выводов на ранних этапах проектирования.

Прежде всего, необходимо понять возможности блоков ввода/вывода для выбранной ПЛИС. В серии 7 все внешние выводы сгруппированы в банки (по 50 выводов в каждом), относящиеся к одному из двух

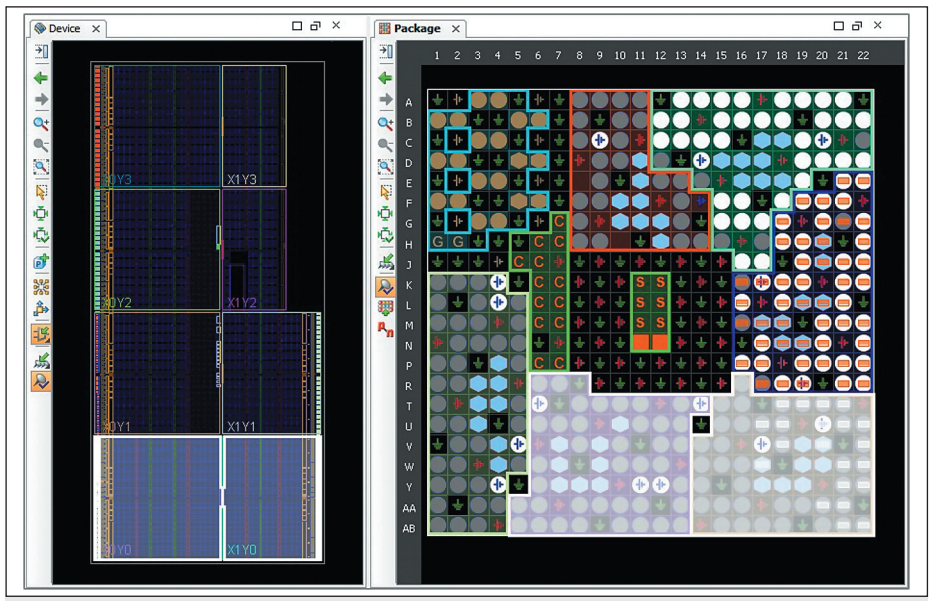

Рис. 3. Совместное использование видов Device и Package в САПР Vivado при планировании подключения внешних сигналов

типов — HR (High-Range) или HP (High-Performance). Блоки High-Range поддерживают более широкий диапазон напряжений питания (до 3,3 В), а High-Performance позволяют достигать более высоких частот передачи сигналов, но ограничены максимальным напряжением 1,8 В.

В таблице приведено соотношение между количеством выводов HR и HP в семействах FPGA серии 7.

Таблица. Соотношение между количеством выводов HR и HP в семействах FPGA серии 7

Семейство	Artix-7	Kintex-7	Virtex-7T	Virtex-7XT Virtex-7HT
High-Range	Все	Большинство	Несколько банков	–
High-Performance	–	Несколько банков	Большинство	Все

Часть выводов в банках ввода/вывода имеет двойное назначение: например, 4 дифференциальные пары, подключаемые к региональным тактовым буферам (BUFR), входы для подключения опорного напряжения (VREF), которое требуется отдельным типам электрических интерфейсов. В определенных банках находятся входы, подключаемые к глобальным тактовым буферам (BUFG), и интерфейс для подсоединения внешней памяти BPI или SPI (его выводы станут пользовательскими после завершения загрузки конфигурации). Если распределять выводы ПЛИС без учета этих особенностей, может оказаться, что какие-то возможности окажутся заблокированными, поскольку соответствующие выводы уже заняты другими сигналами. В частности, если разработчик предполагает применение так называемых source-synchronous-интерфейсов, удобнее использовать региональные тактовые буферы BUFR. Однако для этого нужно, чтобы вход BUFR находился в том же банке, что и все линии данных, которые он будет тактировать.

Кроме того, для достижения максимальных рабочих частот бывает важно взаимное расположение компонентов на кристалле и их положение относительно тех внешних выводов, которые обеспечивают интерфейс с внешними компонентами. Какие-то параметры могут оказаться объективно недостижимыми, если сигналы будут подведены к ПЛИС неудачно, и для поддержания связи между ними придется проводить длинные линии по кристаллу.

Для анализа возможности размещения выводов совместно с их положением на кристалле удобно пользоваться визуальным интерфейсом: виды Device и Package в САПР Vivado показывают физическое расположение ресурсов на кристалле ПЛИС и корпус с отмеченными сигналами (рис. 3). В интерфейсе Vivado реализован так называемый кросс-выбор (cross-selection): если выбрать компонент на одном из представлений, его положение будет показано и на другом представлении. Таким образом, разработчик может убедиться, что компоненты, которые он полагает расположенными рядом, в действительности располагаются именно так. Следует обратить внимание, что два выделенных белым банка ввода/вывода, находящиеся по нижнему краю на рис. 2 (вид Package), соответствуют двум нижним регионам ПЛИС. Эти регионы, например, не могут связать свои блоки DSP48 выделенными трассировочными ресурсами, поскольку для такой операции они должны соседствовать друг с другом по вертикали, а не по горизонтали.

Все это означает, что уже на ранних этапах проектирования изделия на базе ПЛИС необходимо обратить внимание на распределение внешних сигналов по банкам ввода/вывода, избегая решений, впоследствии способных вызвать проблемы с трассировкой ответственных сигналов.

Планирование системы синхронизации

Правильное построение системы синхронизации в современных FPGA не является сложным процессом, однако данный этап не следует оставлять без внимания. Методика выбора системы синхронизации базируется на двух последовательно пришедших в микроэлектронную область подходах:

1. Полностью синхронный стиль проектирования.
2. Глобально асинхронные, локально синхронные схемы.

При этом второй подход не отменяет, а расширяет первый, поскольку «глобальная асинхронность» здесь не означает, будто в каких-то случаях правилами построения синхронных схем можно пренебречь.

Итак, синхронная схема обладает следующими признаками:

- один тактовый сигнал, один перепад (все триггеры используют только фронт или только спад тактового сигнала);
- используются D-триггеры (не защелки);
- регистры на выходах блоков;
- используются сигналы «разрешение счета» вместо управления тактовым сигналом;
- используются схемы синхронизации для асинхронных сигналов.

Не используются:

- тактовые сигналы, полученные с помощью логических вентилей, комбинирования разрядов счетчиков или делением частоты с помощью триггеров логических ячеек;
- локальные асинхронные сигналы сброса/установки.

Для того чтобы тактовый сигнал был доставлен на все синхронные компоненты ПЛИС в одно и то же время, недостаточно просто завести его на один из тактовых буферов. Требуемая схема включения блока формирования тактовых сигналов в его простейшем варианте показана на рис. 4. Генерация и настройка такого блока довольно легко осуществляется с помощью инструмента Core Generator, имеющегося в составе средств разработки Xilinx.

Для того чтобы проиллюстрировать актуальность использования такой схемы, можно рассмотреть взаимное расположение компонентов, представленных на рис. 4, на реальном кристалле FPGA (рис. 5). Если на принципиальной электрической схеме короткий сигнал обратной связи CLKFBIN создает впечатление компактности всего узла, то на рис. 5 видно, что два соединенных им компонента находятся в различных частях кристалла. Это сделано для того, чтобы сигнал обратной связи мог пройти по кристаллу приблизительно такое же расстояние, что и тактовые сигна-

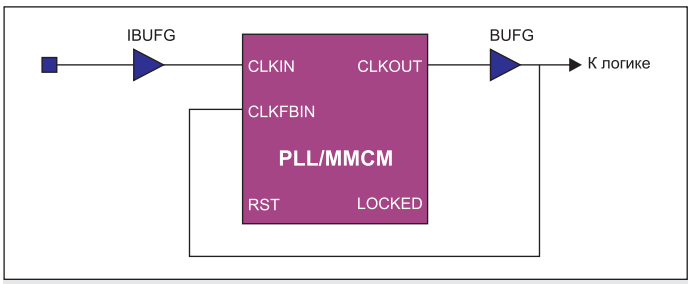

Рис. 4. Схема включения компонента формирования тактового сигнала

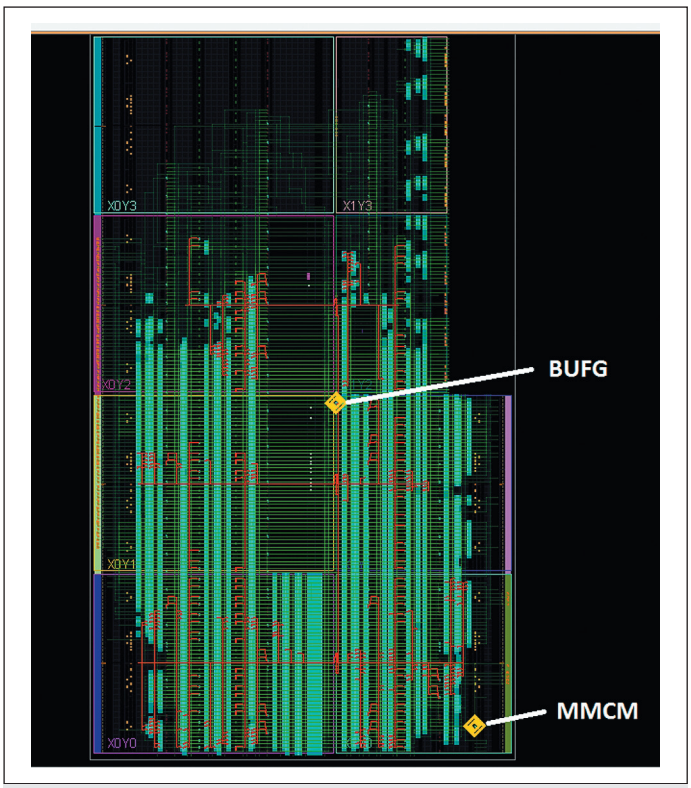

Рис. 5. Взаимное расположение компонентов MMCM и BUFG на кристалле FPGA

лы, подаваемые на синхронные компоненты FPGA (одна из тактовых сетей выделена на рис. 5 красным цветом). Получив фронт сигнала обратной связи, компонент MMCM будет подстраивать его таким образом, чтобы синхронизировать с опорным сигналом CLKIN. Поэтому при наличии в системе синхронизации компонентов MMCM или PLL можно говорить, что все синхронные компоненты проекта получают фронт тактового сигнала в одно и то же время.

Разумеется, абсолютной идентичности в распространении тактового сигнала добиться нельзя. Можно только вести речь о том, что неравномерность его распространения с блоками MMCM/PLL сведена к разумному минимуму, который позволяет говорить о схеме как о полностью синхронной. Рассинхронизация будет увеличиваться по мере возрастания размеров кристалла, и для больших FPGA становится актуален подход GALS (Globally Asynchronous, Locally Synchronous). Он подразумевает, что проект состоит из нескольких синхронных подсистем, работающих каждая на своей тактовой частоте. Номинально эти частоты могут быть и одинаковы, но если каждый тактовый сигнал формируется своим блоком MMCM, то на практике фазы таких сигналов все равно будут отличаться в разных областях кристалла. А потому необходимо заранее предусматривать способы передачи данных из одного тактового региона в другой.

Семейство UltraScale изначально не предполагает формирование общих тактовых сетей, которые могли бы распространяться на весь

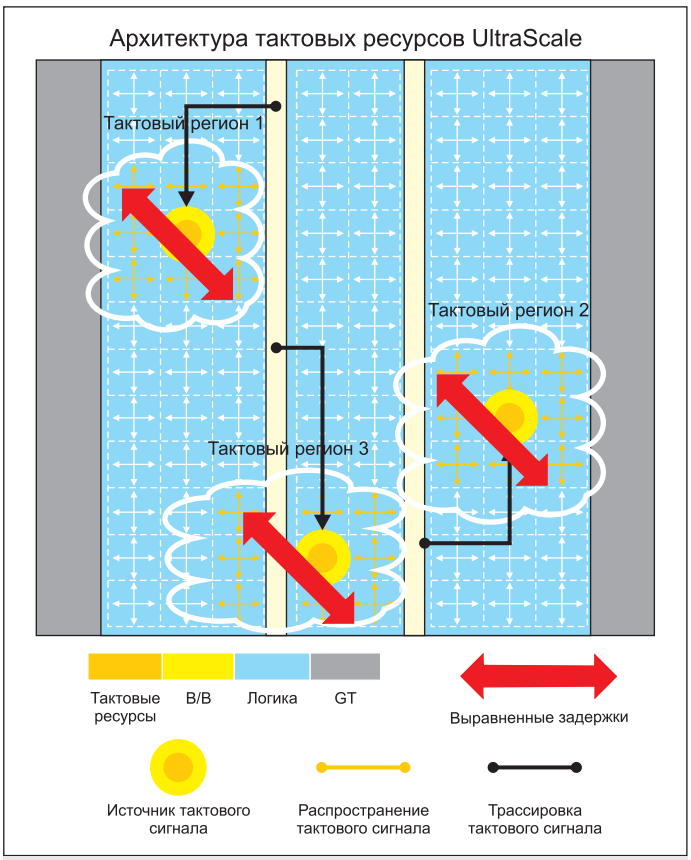

Рис. 6. Тактовые регионы в ПЛИС семейства UltraScale

кристалл. В этом семействе каждый тактовый регион имеет свои модули для формирования стабильного в пределах региона тактового сигнала (рис. 6). Даже если на оба модуля подана одна и та же частота, каждый из модулей будет подстраивать фазу индивидуально, поэтому рано или поздно их выходные сигналы окажутся рассинхронизированы.

Для ПЛИС серии 7 планирование нескольких независимых тактовых регионов пока не является технической необходимостью, однако может быть настоятельно рекомендовано. Это позволит избежать длительных процессов отладки на поздних этапах разработки проекта.

Разделение проекта на тактовые подсистемы производится различными способами — например, по функциональному признаку или по отдельным каналам. В первом случае в проекте формируются такие подсистемы, как цифровые фильтры, коммуникационные и процессорные модули, каждая из которых использует собственный тактовый сигнал. Передача данных между подсистемами выполняется через двухпортовую блочную память, имеющую физически независимые порты, в том числе обеспечивающие два независимых тактовых входа.

Для устройства с большим количеством независимых и параллельно работающих каналов может оказаться полезным выделение собственного тактового сигнала для канала (или группы каналов).

Продолжение следует

Литература

1. Тарасов И. Методология проектирования для ПЛИС Xilinx: организационные аспекты // Компоненты и технологии. 2015. № 1.
2. Тарасов И. Проектирование для ПЛИС Xilinx на языке System Verilog в САПР Vivado // Компоненты и технологии. 2014. № 12.
3. Vivado Design Suite User Guide. Partial Reconfiguration http://xilinx.com/support/documentation/sw_manuals/xilinx2014_4/ug909-vivado-partial-reconfiguration.pdf