УДК 004.047

Методика верификации межсоединений на базе унифицированной тестовой инфраструктуры

И.А. Медведев, Ф.М. Путря

ОАО НПЦ «ЭЛВИС», imedvedev@elvees.com

***Аннотация* — Одним из ключевых компонентов современных систем-на-кристалле (СнК), от которых зависит их производительность, является внутреннее межсоединение. Его функциональная верификация требует больших затрат на создание инфраструктуры, генерацию тестовых воздействий и их отладку, разработку модели и оценку эффективности исполнения целевых задач системы. В статье представлена методика верификации межсоединений, построенная на принципах унификации и повторного использования, обеспечивающая эффективность функциональной верификации и контроль качества работы (Quality of Service) как в автономных тестовых окружениях, так и в составе СнК.**

***Ключевые слова* — межсоединение, система-на-кристалле, сеть-на-кристалле, верификация, UVM, TLM, SystemC.**

I. Особенности верификации межсоединений

В современных СнК коммутационная логика становится специализированным блоком, архитектура которого прорабатывается, исходя из требований целевой задачи, возможностей его разводки и параметров, получаемых на этапе разработки топологии системы. Существенная часть интегрированных на кристалле IP-блоков может быть приобретена у сторонних фирм и иметь специфические интерфейсы взаимодействия с системой. В результате даже в рамках нескольких проектов одной фирмы имеется многообразие топологий и архитектур межсоединений. В рамках одной СнК они комбинируются в различные варианты, начиная от шинной организации и заканчивая сетями-на-кристалле [1], обладающие широкой номенклатурой интерфейсов (AMBA APB/AHB/AXI, OCP, VCI и пр.) для обеспечения связи между различными блоками.

Одним из самых трудоёмких процессов верификации является отладка. А именно обнаружение, локализация и исправление ошибок, которые могут содержаться в аппаратуре, тестовых воздействиях и окружениях. Особую роль играет обнаружение критического блока, ограничивающего производительность системы. Им не обязательно оказывается коммутационная логика, однако анализ характеристик потоков, проходящих через ее интерфейсы, существенно облегчает поиск. Если же критическим блоком оказывается межсоединение, необходим анализ критического места среди основных узлов коммутации (дешифрация адреса ведомого устройства, логика арбитража, разрешение конфликтов, буферизация, виртуальные каналы и пр.). Можно выделить следующие важные аспекты верификации межсоединений:

1) проверка корректности обмена данными с учётом заданной карты памяти;

2) локализация ошибок;

3) оценка качества работы;

4) локализация узких мест в организации СнК или межсоединении в случае неудовлетворительного качества передачи;

5) ускорение процесса создания верификационной инфраструктуры для межсоединений с различной номенклатурой интерфейсов и архитектурой.

В следующих главах рассматриваются подходы к созданию генераторов входных воздействий и эталонных моделей на основе унифицированной инфраструктуры, позволяющие создавать типовые тестовые сценарии и локализовать ошибки в произвольных межсоединениях. Вводится набор метрик и средства их анализа, обеспечивающие предварительный анализ и локализацию проблем качества передачи данных для коммутационных структур, в том числе и в составе СнК.

II. Выявление общих особенностей существующих межсоединений

Исторически можно выделить три поколения межсоединений [2]. На ранних этапах развития СнК использовали стандартную шину для соединения IP-блоков между собой. С эволюцией технологий и с увеличением количества узлов коммуникации стали использоваться сложные шины, организованные иерархическим образом [3]. При этом шинные архитектуры характеризуются низкой степенью масштабируемости. При увеличении числа подключенных устройств возрастает время доступа к шине. Параллельно с этим стали использо-

ваться коммутаторы, организованные по принципу каждый с каждым [4], пропускная способность которых и уровень распараллеливания доступа к ресурсам существенно выше. Однако в дальнейшем сложность реализации, возрастающая с добавлением все большего количества узлов, привела к появлению масштабируемых сетей-на-кристалле [5]. На сегодняшний день в различных по типу и назначению СнК можно встретить любое из представленных межсоединений или их комбинацию. На основе анализа их построения было выявлено, что любое межсоединение:

1) имеет в составе хотя бы один ведомый и хотя бы один ведущий интерфейсы;

2) реализует функцию передачи данных от ведущих к ведомым с возможным обратным ответом.

Следовательно, верификационная инфраструктура должна выполнять:

1) имитацию работы ведущих и ведомых устройств;

2) функциональную проверку;

3) сбор статистики для анализа качественных характеристик работы.

Реализация тестовой инфраструктуры не зависит от средств верификации, в качестве которых в данной статье будут использоваться языки SystemVerilog с библиотекой UVM и C++ с библиотекой SystemC. Их возможности продемонстрированы в статье [6].

III. Абстракция данных

Были сформулированы требования к абстрактной транзакции, содержащей в себе полный набор информации для передачи данных. Будучи универсальной по отношению к любому накристальному интерфейсу, транзакция должна иметь адрес ведомого устройства, к которому доставляется, хранить передаваемые данные, знать их объем и идентифицировать тип обращения (запись либо чтение). В библиотеке UVM определен класс uvm_tlm_generic_payload, удовлетворяющий всем сформулированным требованиям к универсальной транзакции. Его описание также представлено в стандарте TLM-2.0 библиотеки SystemC и принято одной из ведущих организаций в области системного проектирования, моделирования и верификации Accellera System Initiative. Использование такого представления данных позволяет вести разработку тестового окружения, совместимого с любыми другими верификационными компонентами или моделями, предлагающими соответствующий стандартизованный механизм коммуникации. В частности, в рамках данной работы была реализована связь компонентов UVM библиотеки с SystemC моделью.

IV. Имитация работы ведущих и ведомых устройств

Для формирования входных тестовых воздействий был определен новый более высокий уровень абстракции данных в виде потоков данных. Процесс генерации в тестовом окружении сводится к трем уровням (от более к менее высокому):

1) уровень потока данных, наиболее удобный для управления сценариями нагрузки;

2) уровень универсальной транзакции, обеспечивающий унифицированный способ коммуникации между компонентами тестового окружения и межсоединениями, имеющими широкую номенклатуру интерфейсов;

3) интерфейсный уровень, описывающий специализированные особенности представления данных конкретного интерфейса.

Переход между уровнями реализован в рамках иерархического подхода (layering) библиотеки UVM. Ведущие устройства представляются в виде генератора параллельных динамически изменяющихся потоков, конфигурируемых пользователем. Представление данных в виде потоков позволяет имитировать реальную загрузку на интерфейсах межсоединения. Управляя плотностью потоков, способами упаковки данных, определением рабочих диапазонов адресов формируется различный специализированный трафик данных, требования к которому были описаны в первой части статьи. Такой механизм позволяет вскрывать узкие места межсоединений, исследовать их работу при максимальной входной загрузке или воссоздавать краевые условия, присущие обмену данными в текущей архитектуре.

Большая часть параметров входных воздействий может быть описана на самом верхнем уровне абстракции — уровне потоков. Основной набор тестовых сценариев, нацеленный на выявление типовых проблем межсоединений, не зависит от особенностей используемых интерфейсов и организации межсоединений и может быть без изменений повторно использован на следующих проектах. Специфика конкретных интерфейсов или узкоспециализированные тесты, например тестирование корректности логики арбитража, может потребовать дополнительной проработки, однако основной пласт тестов и каркас окружения остаётся неизменным. Такой подход позволяет в процессе верификации уделить больше времени исследованию и сравнительному анализу различных вариантов коммутации.

Ведомое устройство может быть описано в виде модели памяти, отображающей определенные диапазоны адресов заданной карты. Ведомое устройство работает непосредственно с универсальной транзакцией и формирует при необходимости ответные данные, имитируя реальное время доступа и возможные сбои.

V. Функциональная верификация

Проверка корректности работы межсоединения посредством тестового окружения может быть реализована несколькими способами. В первую очередь разработчик тестов может, опираясь на механизмы двунаправленной передачи потоков данных, сравнивать корректность их прихода. А именно, упорядочивая записи и чтения выполнять последующую сверку прочитанных и сформированных для записи данных. Такой подход эффективен в некоторых случаях, но накладывает ограничение на структуру тестовых потоков и порядок их исполнения. Также накладываются ограничения на генерацию случайных воздействий. Второй способ заключается в реализации одного из вариантов эталонной функциональной модели, поддающихся следующей классификации.

По *уровню абстракции данных*:

1) побитовая – полностью имитирует работу устройства с точностью до каждого сигнала (требует переработки под каждый проект, обеспечивает максимальную точность сравнения при одинаковой сложности реализации с аппаратурой);

2) транзакционная – работает на уровне транзакций (менее точная при существенно упрощенной реализации).

По *временному отношению к обработке данных* (или транзакций в случае транзакционных моделей):

1) потактовая – данные формируются точно в такт с реальной схемой (требует переработки под каждый проект, крайне сложная реализация при высокой точности сравнения);

2) последовательностная – данные формируются в той же последовательности, что и в аппаратуре (требует переработки под каждый проект, достаточная точность, использование без ограничений при написании тестовых воздействий);

3) псевдо-последовательностная в рамках временных окон – формирование данных в пределах временных окон (возможно универсальное использование, требует дополнительных механизмов сравнения, проще в реализации);

4) без контроля последовательности – выполняет формирование выходных данных сразу по их приходу на входы в соответствии с картой памяти (самая простая реализация, накладывает ограничения при написании тестов).

Побитовые, потактовые и последовательностные модели являются самыми сложными по реализации и требуют адаптации под каждое новое межсоединение. Их использование не накладывает никаких ограничений на формирование тестовых воздействий. При этом локализация ошибок выполняется с достаточно высокой точностью. Такие модели не являются подходящими для унифицированного тестового окружения в силу архитектурных различий коммутационной логики в разрешении конфликтов при передаче транзакций на один и тот же порт и возможном их переупорядочивании.

Универсальными качествами обладает модель, позволяющая отслеживать набор транзакций, приходящих на ведомые устройства по адресам, к которым они обращены. Сохраняя такие наборы в рамках заданного временного окна можно динамически получать факты прохождения транзакций, явным образом не гарантируя их последовательность. Самую простую с точки зрения реализации модель, не контролирующую последовательность прихода транзакций, можно реализовать исходя из принципа, что любое корректное обращение рано или поздно дойдет до адресата. Использование такой модели не накладывает каких-либо ограничений на написание тестовых сценариев. По результатам моделирования выполняются сравнения памятей, заполненных моделью с одной стороны и обращениями от межсоединения – с другой. Так проверяются ключевые особенности функционирования. Локализация ошибок осуществляется подборкой тестовых сценариев под конкретный функциональный узел.

В рамках данной работы была реализована эталонная TLM модель на SystemC без контроля последовательности транзакций, интеграция которой в тестовое окружение была выполнена с помощью библиотеки ML_UVM, предлагаемой САПР Cadence. Модель работает параллельно с тестируемым RTL-описанием межсоединения, получая тот же поток транзакций типа uvm_tlm_generic_payload от ведущего UVM агента «на лету». Порты компонентов модели типа sc_export и sc_port подключаются соответственно к uvm_blocking_put_port и uvm_blocking_put_imp портам агентов тестового окружения. Соединения обеспечивают функции библиотеки ml_uvm::connect() в тестовом окружении и ml_uvm_register из SystemC библиотеки. Логика межсоединения реализована в модели на уровне TLM-2.0 на основе сокетов (multi_passthrough_sockets), отправляющих транзакции между собой согласно заданной карте памяти маршрутизации. Возможности, которые заложены в механизмы их функционирования, позволяют реализовать большой набор различных по своему типу моделей. В простейшем случае есть возможность гарантировать запись данных в соответствующие ведомые устройства и возврат прочитанных данных в ведущие.

По результату моделирования в каждом UVM компоненте, соответствующему определенному интерфейсу межсоединения, заполняются две выделенные под эти цели памяти результатами активности тестируемого объекта и модели. По окончанию теста происходит сверка таких памятей. Даже в случае простейшей реализации модели обнаруживаются все ошибки, вызванные не прохождением какой-либо транзакции и некорректными записью или чтением данных. Таким образом, гарантируется проверка сформулированного выше функционального назначения любого межсоединения. Немалую роль играет полнота и качество составленных тестовых воздействий в виде потоков данных.

Создание характерных последовательностей, направленных на проверку определенных узлов межсоединения (арбитража, декодирования, буферизации, виртуальных каналов, маршрутизации и пр.) позволяет не только выявлять ошибки, но и в определенной степени локализовать их. Этого зачастую достаточно разработчику для понимания направления поиска неисправности и дальнейшей детализации тестовых воздействий, направленных на проверку конкретного архитектурного блока межсоединения.

VI. Сбор и анализ статистики

Описанный выше механизм генерации универсальных потоков требует качественного подхода к анализу производительности межсоединений. Для этого, помимо функциональной проверки, существует необходимость разработки специализированных механизмов сбора информации, систематизации метрик для анализа эффективности работы верифицируемых объектов и визуализации полученных результатов. В дальнейшем подразумевается обработка полученных данных в целях выявления критических мест, неэффективности использования ресурсов и предъявления рекомендаций по доработке функциональных возможностей. Нами был разработан механизм сбора статистики, обрабатывающий времена начала и конца передач транзакций, их размер и направление. Представлен комплекс метрик, вычисляемых на основе полученной статистической информации, для системного анализа эффективности передачи данных.

A. Метрики

При формировании и апробировании набора оценочных метрик, были учтены некоторые важные особенности.

Во-первых, учитывая способ сбора статистики на уровне транзакций при имеющейся информации об объеме данных в транзакции, существует возможность раздельного анализа статистики: с одной стороны – с точки зрения передачи транзакций, с другой – передаваемых данных. Такой независимый анализ позволяет абстрагироваться в некоторых случаях от конкретных интерфейсных особенностей, касающихся способов представления и упаковки данных в транзакциях, что может быть полезно разработчику блока межсоединения при его проектировании и отладке. В других случаях появляется возможность анализировать работу межсоединений в стандартизованных единицах измерения (байты и секунды).

Во-вторых, обеспечение независимости интерфейсов друг от друга по используемому тактовому сигналу. Так при работе портов на разных частотах в рамках одного межсоединения необходимо обеспечивать единый для всех временной отсчет времени. В противном случае, привязываясь к конкретному таковому сигналу, совместный анализ характеристик работы по разным интерфейсам будет некорректен. С другой стороны, в процессе проектирования устройства удобно проводить анализ метрик, временная шкала которых измеряется в тактах, а не в стандартизованных величинах (например, в секундах). Тот и другой варианты реализации могут быть использованы при сборе статистики.

Учитывая все рассмотренные факторы, был сформулирован набор метрик, отражающий характер функционирования межсоединения:

1) среднее время передачи транзакций;
2) максимальное время передачи транзакций;
3) среднее время передачи данных;
4) максимальное время передачи данных;
5) средний объем данных за транзакцию;
6) максимальный объем данных за транзакцию;
7) средняя скорость передачи транзакций;
8) максимальная скорость передачи транзакций;
9) скорость передачи транзакций на канал;
10) средняя скорость передачи данных;
11) максимальная скорость передачи данных;
12) скорость передачи данных на канал.

B. Использование метрик для анализа эффективности работы межсоединений

Существует несколько направлений использования собираемой статистики. В первую очередь это анализ производительности всей системы, исходя из мониторинга потоков данных на межсоединении. На рис. 1 представлен пример СнК. Важной задачей является оценка исполнения определенных целевых задач на всей системе.

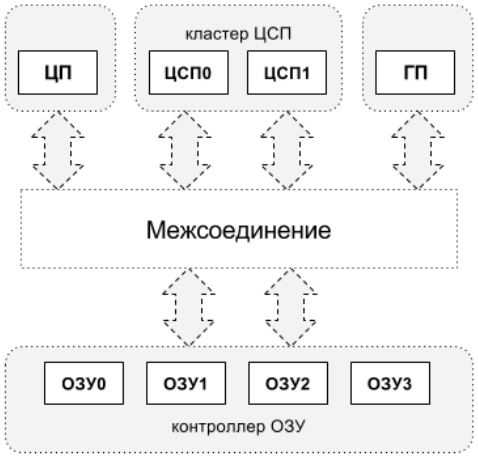

Рис. 1. Пример верифицируемой СнК

Подключение в тестовое окружение пассивных компонентов (мониторов, сборщиков статистики) на внутренние интерфейсы СнК (рис. 2) позволяет в процессе исполнения целевой задачи собирать информацию об активности процессов передачи данных.

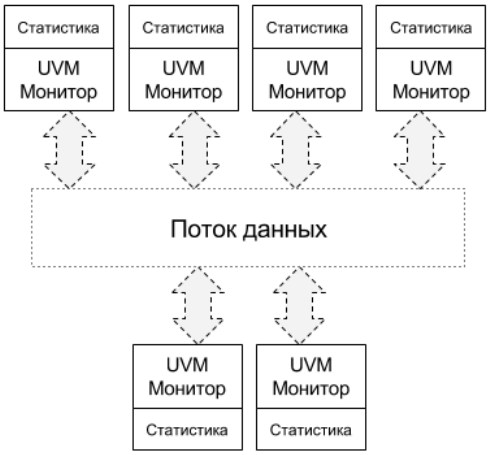

Рис. 2. Мониторинг и сбор статистики при верификации СнК на глобальном уровне

Сформулированный набор метрик дает возможность производить первичную оценку эффективности работы всей системы. Используя автоматизированные аналитические возможности (например, определение моментов снижения скорости передачи данных на интерфейсе ниже его пропускной способности), а также визуализацию метрик, формируются выводы о качестве архитектурной реализации под конкретную задачу, об эффективности написанной программы, об узких местах СнК и в том числе о компонентах, тормозящих исполнение алгоритма (медленная работа памяти, неэффективность межсоединения, блокировка ведущих ядер и др.). Такой подход не требует глубоко детализированных оценочных метрик, разработан под каждое устройство и опирается именно на общую характеристику потока данных, покрываемую представленными в работе метриками.

Второе направление реализуется при локальной верификации межсоединения, где важно подтвердить заявленные характеристики производительности. Часто разработчик имеет возможность в процессе создания или доработки RTL-описания опираться на анализ собранной статистики для выбора направления модификации. Расширяя набор тестовых воздействий, сформированных при функциональной верификации, в целях создания жесткого использования определенных ресурсов межсоединения обрабатывается статистическая информация о работе устройства в виде предложенных метрик. Далее делается вывод о необходимости внесения изменений либо в конкретную реализацию конкретной функциональной части, либо в общую архитектуру, либо в техническое задание в случае невозможности исполнения требований. При этом унифицированная структура тестового окружения позволяет одни и те же тестовые потоки исполнять на различных межсоединениях и их реализациях, получать статистические результаты в виде стандартизованных метрик и делать сравнительные выводы об эффективности работы. В целях визуализации собираемой статистики была написана программа на языке Python средствами библиотеки matprotlib.

C. Пример использование метрик для анализа

В качестве наглядных примеров были смоделированы в упрощенном виде ситуации, возникающие в реальных проектах. Рассмотрим межсоединение с двумя ведущими (M0, M1) и одним ведомым интерфейсами.

Ситуация 1 описывает вариант исследования эффективности арбитража. Допустим на интерфейсе M0 должен приниматься поток данных со средней скоростью не меньшей 0,8 ГБайт/с при пропускной способности канала в 1 ГБайт/с. На интерфейсе M1 параллельно идет обработка другого трафика, не обладающего жесткими ограничениями. В результате сбора статистики получаем следующую картину (рис. 3).

Межсоединение не поддерживает заявленную среднюю скорость обработки данных по интерфейсу M0. Учитывая появление в определенных моментах потока данных по интерфейсу M1, с высокой степенью вероятности можно предположить, что именно он является причиной «проседания» до 0,2 ГБайт/с потока на M0. Логика арбитража как раз и отвечает за взаимодействия потоков с разных ведущих интерфейсов. В результате можно рекомендовать внесение модификаций в его реализацию. При необходимости можно детализировать такую ситуацию, изменив определенным образом характеристики потоков.

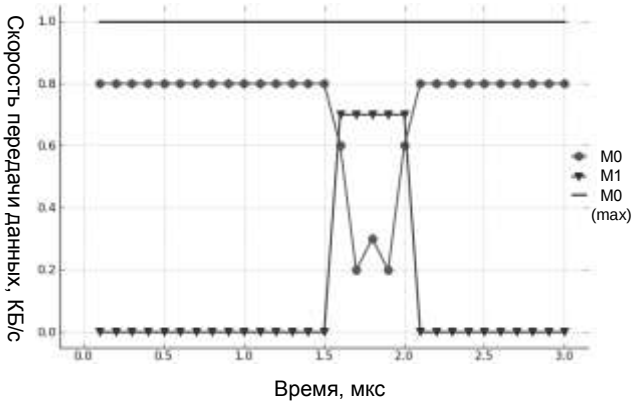

Рис. 3. Средняя скорость передачи данных. Исследование логики арбитража

Ситуация 2. На том же примере можно рассчитать продуктивность использования интерфейса. Например, для M0 эта характеристика будет равна 0,73 % на приведенном интервале от 0 до 3 мкс.

Ситуация 3. Допустим, при уже отлаженном механизме арбитража возникает необходимость обеспечить задержку передачи данных не больше 0,5 нс на байт. На рис. 4 показан пример, в котором два ведущих интерфейса создают поток на одном ведомом. При этом в определенные моменты при возникновении конфликтных ситуаций задержка на интерфейсе M0 превышает

критическое значение. В данном случае разработчику имеет смысл обратить внимание на буферизацию запросов, либо, если это имеет место быть — на реализацию виртуальных каналов, так как подобного рода механизмы отвечают за сохранение обращений в межсоединении.

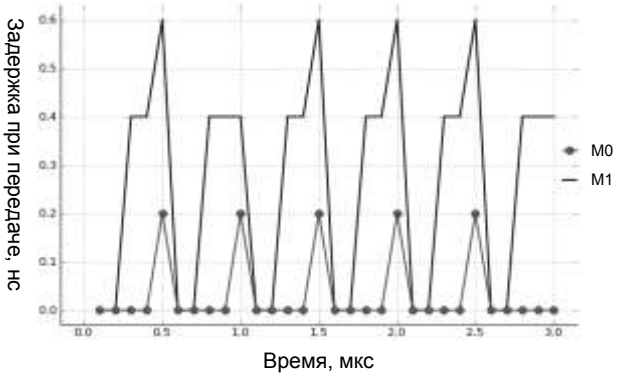

Рис. 4. Максимально время передачи данных. Исследование эффективности буферизации

Очевидно, что в некоторых вариантах формирования потоков данных недостаточно обладать таким набором метрик для четкого определения критического места, особенно при комплексных передачах различного типа. Но итерационный подход по направленному изменению тестовых воздействий либо их дополнению для большей детализации может существенно облегчить поиск.

Рис. 5. Структура универсального тестового окружения

VII. Универсальное тестовое окружение

Описанные методики с учетом абстракции данных позволяют сформировать универсальное тестовое окружение, структура которого показана на рис. 5. Оно было полностью реализовано и использовано для тестирования десятка межсоединений, различных по своей структуре, организации и назначению. Со временем с добавлением новых проектов создается целая библиотека потоков, унифицированная для любой коммутационной логики, обладающая полным набором тестовых сценариев для покрытия критических ситуаций и моделирования трафика реальной загрузки. Все элементы тестовой инфраструктуры отлаживаются и дорабатываются в процессе работы, что положительно сказывается на качестве верификации всех межсоединений.

Применение универсального тестового окружения возможно и в пассивном режиме при подключении в окружение иерархически более высокого уровня. В таком случае отпадает необходимость имитации ведущих и ведомых устройств при сохранении всех отладочных функций в виде сбора и анализа статистики и средств функциональной верификации.

VIII. Заключение

В работе проведен анализ существующих способов организации межсоединений в СнК. Обозначен набор требований к инфраструктуре верификации (к источникам тестовых воздействий, функциональным моделям, средствам сбора и анализа статистики). Определен набор метрик, достаточных для анализа функциональных характеристик коммутатора при различных типах нагрузки. Описаны возможности аналитической обработки статистических данных с формированием рекомендаций по оптимизации соответствующих архитектурных реализаций и разработке требований к обрабатываемому трафику данных. Предложено унифицированное представление межсоединения, на основании которого была введена абстракция данных в виде универсальной транзакции. По результатам этого сформирована структура универсального тестового окружения.

Литература

[1] Bjerregaard T., Mahadevan S. A Survey of Research and Practices of Network-on-Chip // ACM Computing Surveys (CSUR). 2006. № 38, Issue 1.

[2] Kumar R., Zyuban V., Tullsen D.M. Interconnections in Multi-Core Architectures: Understanding Mechanisms, Overheads and Scaling // Proceedings of the 32nd annual international symposium on Computer Architecture. 2005. P. 408-419.

[3] Furber. S. Future trends in SoC interconnect // VLSI Design, Automation and Test, (VLSI-TSA-DAT). 2005. P. 295-298.

[4] Kim D. A reconfigurable crossbar switch with adaptive bandwidth control for networks-on-chip // Circuits and Systems, 2005, (ISCAS) / IEEE International Symposium on, 2005. P. 2369-2372.

[5] Медведев И.А. Анализ эффективности применения буферизации для маршрутизации в сети-на-кристалле // V Всероссийская научно-техническая конференция «Проблемы разработки перспективных микро- и наноэлектронных систем - 2012». Сборник трудов / под общ. ред. академика РАН А.Л. Стемпковского. М.: ИППМ РАН, 2012. С. 445-450.

[6] Путря Ф.М. Особенности использования возможностей объектно-ориентированного программирования SystemVerilog для функциональной верификации многоядерных СнК // V Всероссийская научно-техническая конференция «Проблемы разработки перспективных микро- и наноэлектронных систем - 2012». Сборник трудов / под общ. ред. академика РАН А.Л. Стемпковского. М.: ИППМ РАН, 2012. С. 83-88.