УДК 004.052.42

Автономная верификация микропроцессоров на основе эталонных моделей разного уровня абстракции

В.А. Шмелёв, И.А. Стотланд

ЗАО «МЦСТ», shmelyov_v@mcst.ru, stotl_i@mcst.ru

***Аннотация* — В статье рассмотрены особенности автономной верификации микропроцессоров. Предложена классификация модулей микропроцессоров с точки зрения их функциональной верификации. Обозначены два основных подхода к автономной верификации в зависимости от принадлежности верифицируемого модуля к одному из подклассов. Предложены и обоснованы архитектуры тестовых систем на основе потактовых и функциональных эталонных моделей. Описан опыт практического применения предлагаемой методики.**

***Ключевые слова* — автономная верификация, тестовая система, эталонная модель, конвейер, микропроцессор, OVM, TLM.**

I. Введение

Развитие микропроцессорной техники связано с постоянным усложнением создаваемых систем, которое остро ставит вопрос проверки их правильности, начиная с самых ранних стадий маршрута проектирования. Усилия, затрачиваемые на обеспечение уверенности в корректности системы, поглощают все возрастающую долю стоимости и времени цикла проектирования. Как отмечает Дж. Бергерон [1], на проверку функциональной корректности СБИС требуется около 70% трудозатрат; число инженеров, занимающихся верификацией, обычно вдвое больше числа разработчиков логической модели; размеры тестов достигают 80% объема всего исходного кода проекта.

Исправление аппаратных ошибок сопряжено с очень большими затратами [2], поэтому столь пристальное внимание уделяется процессу верификации микропроцессорных систем. В соответствии с ГОСТ Р ИСО 9000-2001 «*верификация* — это подтверждение на основе предъявления объективных свидетельств того, что установленные требования были выполнены» [3]. На каждом этапе проектирования применяются специализированные средства верификации, призванные не пропустить неисправности на последующие стадии разработки. Один из ключевых параметров разработки микропроцессоров – time-to-market – предполагает активизацию усилий в целях поиска новых методов и средств верификации проектируемых систем. Одним из путей сокращения сроков проектирования микропроцессоров является применение методов автономной верификации отдельных функциональных блоков на самых ранних стадиях разработки.

В настоящее время не существует единой универсальной методологии автономной верификации, учитывающей специфику верифицируемых RTL-моделей (Register Transfer Level). В данной статье предложена классификация функциональных блоков микропроцессоров с точки зрения их верификации, а также методика автономной верификации микропроцессоров, основанная на двух основных подходах к построению тестовых систем.

II. Обзор подходов к верификации микропроцессоров

Чтобы убедиться, что система удовлетворяет всем функциональным требованиям, сформулированным в спецификациях, применяется *функциональная верификация, о*бъектом которой являются логические модели уровня регистровых передач (RTL-модели), написанные на одном из применяемых при проектировании языков описания аппаратного обеспечения (HDL-языки). Варианты функциональной верификации можно разделить на два основных класса: *динамическую верификацию*, основанную на имитационном тестировании (simulation-based verification) и *формальную верификацию* (formal methods).

В [4] под *формальной верификацией* понимаются приемы и методы формального доказательства (или опровержения) того, что модель системы удовлетворяет заданной формальной спецификации. Одним из наиболее распространенных методов является, разработанный Эд. Кларком [5], метод проверки на модели. Формальные методы основаны на исследовании модели аппаратуры в статическом состоянии.

Центральной проблемой формальной верификации является возможный «комбинаторный взрыв» в пространстве состояний (state explosion problem) при увеличении сложности верифицируемых моделей. Это существенно ограничивает применение формальных методов для верификации RTL-моделей промышленных разработок.

Динамическая верификация осуществляется посредством наблюдения за процессом изменения состояний системы во времени. В основе ее методов лежит генерация входных воздействий и проверка правильности реакции на них. Основные преимущества методов динамической верификации заключаются в том, что они:

1) применимы на ранних стадиях разработки, когда проектируемая система содержит большое число ошибок;

2) дают возможность проверить не только спецификацию, но и конкретную реализацию в виде анализа регистровой модели (RTL-модели).

При верификации сложных промышленных RTL-моделей зачастую применяется динамическая верификация на основе тестовых систем. На рис. 1 представлен график зависимости кодового покрытия от времени при верификации ее методами [6].

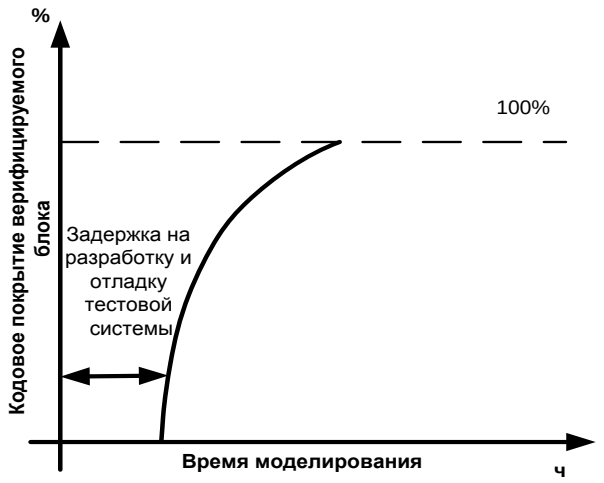

Рис. 1. График зависимости кодового покрытия от времени моделирования для методов верификации на основе тестовых систем

Тестовая система имитирует окружение, в котором будет в дальнейшем функционировать верифицируемое устройство. Среди других задач тестовой системы можно выделить задачи генерации входных последовательностей и проверку правильности выходных последовательностей, а также оценку полноты тестового покрытия.

Сегодня, при всей сложности разрабатываемых систем, актуальна иерархическая структура проектов, когда более сложная система строится на основе более простых блоков. Процесс верификации также основан на принципе иерархичности: существует несколько уровней, на каждом из которых верифицируются структурные элементы всей системы. Выделяют два основных уровня верификации микропроцессоров - *системный* и *модульный*. Верификацию на модульном уровне также называют *автономной* (stand-alone verification). При системной верификации ее объектом является целый микропроцессор. Существует ряд разработанных методов и средств динамической верификации микропроцессорных систем на обоих уровнях. В работах А.С. Камкина [7, 8] рассмотрен метод модульной верификации микропроцессоров на основе формальных спецификаций. Метод, описанный в [8], ориентирован на проверку конвейеризированных модулей микропроцессоров. В работе С. Г. Шаршунова и В. В. Белкина [9] представлен метод функционального диагностирования моделей RISC-микропроцессоров. В ней делается упор на конкретную архитектуру микропроцессора, что существенно ограничивает применимость результатов исследования.

Основным преимуществом автономной верификации является возможность проводить тестирование на ранних стадиях разработки, не дожидаясь спецификации всей системы в целом, как только готова спецификация RTL-модели модуля, входящего в ее состав. На модульном уровне также возможно создание необходимой динамики работы модели для достижения критических условий (переполнение буферов, подача запрещенных входных данных). Центральным вопросом при разработке тестовых систем для автономной динамической верификации RTL-моделей является способ проверки правильности результатов симуляции. Ричард Хо в своей диссертации [10] выделяет два основных способа: самопроверяющие тесты (self-checking) и сравнение с эталонной моделью (co-simulation).

В самопроверяющей или *детерминированной* тестовой системе устройство проверки осуществляет не только анализ выходных сигналов, но и описывает правильную функциональность верифицируемого устройства. Детерминированные тестовые системы требуют больших усилий на разработку и поддержку тестов. При изменении функциональности или интерфейса верифицируемой системы необходима их тщательная переработка.

В связи с этим преимущество имеет применении тестовых систем на основе эталонных программных моделей. Этот подход лишен недостатков, связанных и использованием детерминированных тестовых систем, так как в данном случае отдельно от самой тестовой системы разрабатывается программная модель верифицируемого устройства, обладающая той же функциональностью, но, описанная на более абстрактном уровне.

Существует целый ряд подходов к построению тестовых систем, таких как расширенная методология верификации (*Advanced Verification Methodology, AVM*), открытая методология верификации (*Open Verification Methodology, OVM*), универсальная методология верификации (*Universal Verification Methodology, UVM*). Каждая из этих методологий предлагает некоторый набор библиотечных средств построения тестовых систем и документацию по их применению. Данные подходы задают архитектуру тестовых систем в виде набора компонентов, определенного в библиотеках, сопровождающих каждую методологию. Однако все методологии

предусматривают проверку корректности тестируемой системы с помощью утверждений, поэтому они не касаются вопроса построения эталонных моделей и организации взаимодействия эталонных моделей и тестовой системы.

На основе проведенного исследования существующих подходов к функциональной верификации микропроцессоров можно сделать вывод об актуальности разработки новой методики автономной верификации основе тестовых систем с применением эталонных моделей. Перспективным является использование библиотек одной из стандартных методологий верификации (OVM, UVM) для построения универсальных конфигурируемых тестовых систем. Стоит также отметить, что ни в одном из существующих подходов к автономной верификации нет четких рекомендаций по построению тестовых систем и эталонных моделей в зависимости от архитектуры и специфики верифицируемого модуля.

III. Классификация функциональных блоков микропроцессоров

Несмотря на существенные отличия микропроцессоров с различными архитектурами, они обладают рядом общих для всех архитектур набором модулей. Существует несколько подходов к классификации функциональных блоков микропроцессоров по их назначению, способу организации, архитектуре. Однако, в настоящее время не существует ни одной классификации функциональных блоков, входящих в состав микропроцессоров, с точки зрения их особенностей при функциональной верификации. В данной работе показана попытка такого разделения функциональных блоков, входящих в состав микропроцессоров.

Функциональные блоки, входящие в состав микропроцессорных систем можно разделить на два класса:

1) *Конвейеризированные (операционные) модули* — системы, организующие и поддерживающие процесс вычислений. Как правило, строятся на основе конвейерной архитектуры.

2) *Модули системного обмена* — системы, обеспечивающие взаимодействие между операционными блоками микропроцессора, процессорным ядром и памятью, микропроцессором и внешними абонентами микропроцессорной системы, а также организующие межпроцессорный обмен внутри многопроцессорных и многокластерных систем.

Главной отличительной особенностью операционных модулей является их конвейерная архитектура, что влечет за собой ряд существенных затруднений при построении тестовых систем и разработке тестов. Операционные модули в процессе своей работы изменяют сами передаваемые данные, которые зачастую являются немаркированными.

К основным особенностям модулей системного обмена с точки зрения функциональной верификации относятся:

- непрерывное бесконечное функционирование;
- тот факт, что параметром функции преобразования является только формат представления данных, сами данные остаются неизменными;
- модули осуществляют коммутацию данных;
- маркированность данных (однозначность отображения множества входных воздействий во множество реакций);
- отсутствие конвейера и, как правило, четких временных характеристик обработки транзакций (в тактах);
- асинхронность входных и выходных данных.

Далее в статье рассмотрены основные особенности каждого из выделенных подклассов при их функциональной автономной верификации. Представлены подходы к построению тестовых систем, эталонных моделей и организации их взаимодействия.

IV. Подходы к автономной верификации микропроцессоров

A. *Подход к автономной верификации конвейеризированных модулей*

Конвейеры широко применяются в современной цифровой технике. Для конвейерных устройств характерно жесткое разделение процесса обработки данных или запросов по тактам работы конвейера. С точки зрения функциональной верификации это свойство создаёт ряд проблем:

1) В тестовой системе требуется моделировать работу конвейера соседнего модуля или нескольких модулей, поскольку входные сигналы на устройство необходимо подавать в точном соответствии со спецификацией на временные ограничения конвейера.

2) Для верификации устройства недостаточно проверить его функциональность, необходимо также проверять выдачу выходных сигналов модуля в соответствии со спецификацией конвейера.

3) Применение методологий построения тестовых систем, в которых тестовые воздействия описываются в виде высокоуровневых транзакций (AVM, OVM и т.п.) неэффективно, поскольку запросы, выполняемые на конвейере, не обладают атомарностью. В ходе обработки запроса может возникнуть ситуация, когда необходимо отменить или повторить все транзакции, уже запущенные на конвейере.

Кроме того, указанные выше особенности накладывают ряд требований на способ проверки верифицируемого устройства. Для проверки модулей конвейерной обработки запросов чаще всего

используются эталонные модели, разработанные с потактовой точностью. Их применение позволяет с помощью небольшого набора простых утверждений (assertions) проверить корректность работы устройства. Однако, применение высокоуровневых языков программирования зачастую не позволяет повторить все особенности функционирования модуля, в том числе по причинам, связанным с принципиальными отличиями языков программирования от языков описания аппаратуры [1]. Помимо этого, разработка потактовой эталонной модели требует значительно больших ресурсов по сравнению с функциональной моделью, причем с ростом сложности устройства время разработки растёт нелинейно и его трудно прогнозировать. Этот фактор накладывает ограничения на классы устройств, для которых можно разработать потактовую модель в приемлемые сроки. На рис. 2. представлена архитектура тестовой системы и окружения для верификации конвейеризированного модуля.

Система включает в себя окружение (ENV), содержащее модель конвейера(CONV) и преобразователь входных воздействий в битовый формат (SER), генератор высокоуровневых воздействий (TRANSACTOR), а также модуль проверки (CHECKER), в состав которого входит набор утверждений (ASR) для проверки корректности работы верифицируемого устройства (DUV) и потактовая эталонная программная модель (CMODEL). Между преобразователем входных воздействий и моделью конвейера введена обратная связь для передачи текущего состояния конвейера.

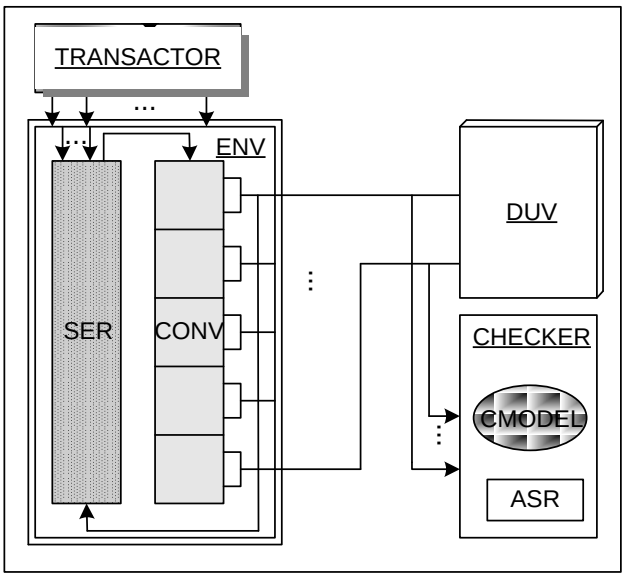

Рис. 2. Тестовая система для верификации конвейеризированных модулей

Введение в тестовое окружение модели конвейера и его обратной связи с преобразователем входных воздействий позволяет решить перечисленные выше проблемы, связанные с временными ограничениями и значимостью данных конвейеризированных модулей.

Корректность данных контролируется набором SystemVerilog утверждений, входящих в состав тестового окружения.

B. Подход к автономной верификации модулей системного обмена

Одними из основных свойств модулей системного обмена является их неконвейеризированная архитектура и отсутствие жестких временных ограничений на обработку транзакции. Благодаря этому, при верификации таких модулей возможен переход с потактового регистрового уровня на уровень транзакций (TLM) с использованием функциональных эталонных моделей. Однако, в таком случае встает вопрос преобразования входов и выходов тестовой системой.

Один из подходов к организации взаимодействия тестовой системы и эталонной модели при переходе на уровень транзакций основан на преобразовании входных воздействий в битовый формат (сериализацию) и выходных реакций в формат транзакций (десериализацию).

Сериализацию можно представить как отображение вида:

$$T : S_{tlm} \times I_{trans} \rightarrow S_t \times I_{bit},$$

где T — отображение входных воздействий в битовый формат;

S_{tlm} — конечное множество состояний автомата обработки единичных воздействий на уровне TLM;

S_t — конечное множество состояний автомата обработки единичных воздействий на потактовом уровне;

I_{tran} — конечное множество входных воздействий в формате транзакций;

I_{bit} — конечное множество входных воздействий в битовом формате.

В свою очередь, *десериализация* — это:

$$D : S_t \times O_{bit} \rightarrow S_{tlm} \times O_{tlm},$$

где D — отображение выходных воздействий в битовом формате в выходные транзакции TLM;

S_t — конечное множество состояний автомата обработки единичных воздействий на потактовом уровне;

S_{tlm} — конечное множество состояний автомата обработки единичных воздействий на уровне TLM;

O_{bi} — конечное множество реакций в битовой формате;

O_{tran} — конечное множество реакций в формате транзакций.

При построении тестовой системы с применением методологии OVM преобразование входных стимулов осуществляют *драйверы* тестового окружения, преобразование реакций — *мониторы*.

На рис. 3 представлена архитектура тестовой системы для верификации модулей системного обмена. В отличие от схемы для конвейеризированных модулей данная тестовая система основана на эталонной модели уровня TLM, что определяет наличие в ней преобразователей входных воздействий (SER) и реакций (DES) и дополнительного компаратора реакций (COMP) в устройстве сравнения (CHECKER). Компаратор реакций имеет в своем составе набор буферов, хранящих реакции модели. В компараторе реализуется функция поиска транзакции, полученной от верифицируемого устройства, в соответствующем буфере транзакций эталонной модели. Аналогично подходу, описанному в пункте A, транзакции уровня TLM создаются генератором высокоуровневых транзакций (TRANSACTOR).

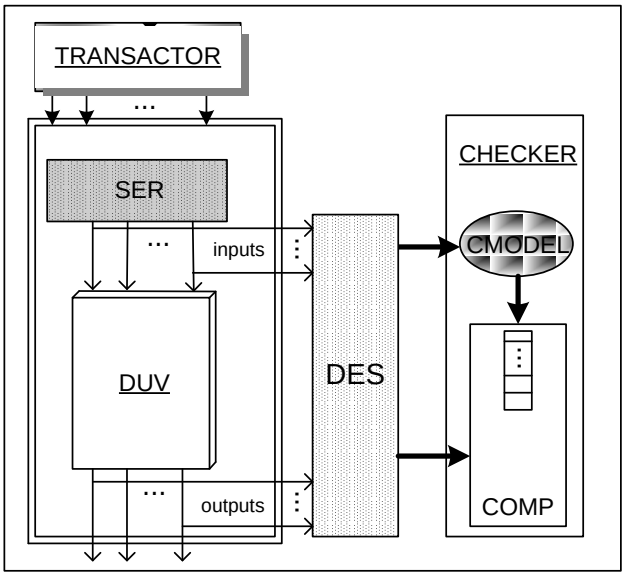

Рис. 3. Тестовая система для верификации модулей системного обмена

V. Опыт практического применения

Описанные в части IV подходы были применены при автономной верификации функциональных модулей микропроцессора «Эльбрус-2S», разрабатываемого в ЗАО «МЦСТ». В зависимости от принадлежности верифицируемых модулей к одному из подклассов, представленных в части III, были разработаны тестовые системы с архитектурами, показанными на рис. 2 и рис. 3, первая из которых характеризует процесс верификации буфера записи (STMB), как примера конвейеризированного модуля, а вторая - хост-контроллера (Host_ctrl) микропроцессора «Эльбрус-2S», являющегося модулем системного обмена.

A. *Верификация буфера записи STMB*

Буфер записи STMB входит в состав модуля кэш-памяти второго уровня (L2) микропроцессора Эльбрус-2S. Буфер реализует различные варианты операций записи в память, обработку запросов поддержания когерентности, атомарные последовательности операций. Устройство имеет три конвейерных канала обработки запросов — канал обработки процессорных запросов D (стадии конвейера D1-D3), канал обработки когерентных запросов S (стадии конвейера S3-S6) и канал приёма ответов по чтению из системы M (стадии конвейера M1-M4). Часть входов устройства используются каналами D и S совместно. Кроме того STMB имеет ряд неконвейеризованных интерфейсов.

В качестве общего подхода к верификации устройства используется методика «серого ящика» — воздействия в тестах генерируются с учётом внутреннего состояния буфера. STMB является конвейеризированным устройством, поэтому для верификации модуля было решено использовать архитектуру тестовой системы, представленную на рис. 2. В качестве основы для реализации тестового окружения ENV применены компоненты из библиотеки OVM. Это позволило использовать возможности OVM для конфигурирования тестового окружения и облегчило написание коротких направленных тестов, необходимых на ранних этапах разработки окружения и модели. В состав модуля тестового окружения входят модель конвейера, основанная на очередях, имитирующих конвейеры каналов D, S и M и очередях повторных и вторичных запросов. Кроме того, модель конвейера реализует функции для синхронизации значений на входах и выходах устройства и функцию, вызываемую для моделирования одного такта системного синхросигнала.

Для длинных случайных тестов с ограничениями был реализован генератор входных воздействий на основе технологии C++TESK без аппаратных расширений [8]. Он содержит набор функций-стимулов, которые задают все возможные воздействия на верифицируемый модуль, и функции, накладывающие ограничения на запуск воздействий. Источниками этих ограничений служат как спецификации на устройство и окружающие модули, так и спецификации системного уровня. Постусловия для всех стимулов тривиальные, так как проверка правильности работы устройства выполняется отдельным модулем. Взаимодействие с генератором входных воздействий осуществляется по интерфейсу DPI SystemVerilog (Direct Programming Interface). Устройство проверки организует взаимодействие с потактовой эталонной моделью и содержит набор утверждений (assertions), сравнивающий для значимых запросов выходные порты с эталоном. В составе устройства проверки также имеется дополнительный набор утверждений, проверяющий корректность подаваемых воздействий.

Разработанная таким образом тестовая система позволила покрыть тестовыми сценариями функциональность, указанную в спецификации на устройство.

B. Верификация хост-контроллера

Хост-контроллер организует взаимодействие между процессором и контроллером периферийных устройств (IOHUB), конструктивно расположенном на отдельном кристалле. Хост-контроллер преобразует запросы от процессоров в запросы, понятные IOHUB, формирует соответствующие пакеты и передаёт их в каналы ввода-вывода. При поступлении пакетов запросов от IOHUB (DMA-запросы), хост-контроллер преобразует их в соответствующие запросы к системной памяти и отслеживает их выдачу и выполнение. Таким образом, хост-контроллер можно отнести к модулям системного обмена.

При его автономной верификации был применен подход, описанный в пункте B части IV данной статьи. Тестовое окружение реализовано с использованием компонентов библиотеки OVM. На основе методологии OVM построены генератор высокоуровневых транзакций, преобразователи входных воздействий и реакций, устройство сравнения. Так как хост-контроллер является модулем системного обмена, не имеет жестких временных ограничений обработки транзакций и характеризуется маркированностью транзакций (поля label и nreg), для верификации была использована функциональная эталонная модель уровня TLM.

Эталонная модель принимает входные транзакции и немедленно выдает выходные реакции, которые буферизуются в устройстве сравнения тестового окружения. Выходные реакции верифицируемого устройства сначала преобразуются в формат транзакций мониторами тестового окружения, после чего запускается функция поиска полученных транзакций в соответствующих буферах эталонной модели. Применение буферизующего устройства сравнения является отличительной особенностью тестовых систем модулей системного обмена, характеризующихся маркированностью данных. Благодаря его использованию, нет необходимости в реализации сложной внутренней организации хост-контроллера (внутренних арбитров и обратных связей исходной RTL-модели) и учета временных характеристик устройства. Такой подход существенно облегчил и ускорил процесс разработки эталонной модели. При автономной верификации хост-контроллера был обнаружен ряд ошибок в RTL-модели и спецификациях, связанных с преобразованием форматов данных, обработкой очередей запросов, упорядоченностью данных.

VI. Заключение

Описанные в статье подходы к автономной верификации модулей микропроцессоров положены в основу комплексной методики автономной верификации, ключевой особенностью которой является учет функциональных и архитектурных характеристик верифицируемых модулей. По сравнению с универсальными подходами, основным преимуществом подхода к верификации конвейеризированных устройств является простота реализации устройства сравнения и возможность анализа внутреннего состояния конвейера, а преимуществом подхода к верификации модулей системного обмена - существенное упрощение эталонной модели путем перехода на уровень транзакций.

Опыт практического применения принятой методики при автономной верификации модулей микропроцессора «Эльбрус-2S» показал ее эффективность. В процессе работы были выявлены ряд проблем, связанных с организацией взаимодействия потактовых моделей и тестового окружения и требованиями к спецификациям устройств. Дальнейшими направлениями представленного исследования будут формализация спецификаций, автоматизация построения тестовых систем и эталонных моделей.

Литература

[1] Bergeron J. Writing testbenches: functional verification of HDL models. Kluwer Academic Publishers. 2003. 479 p.

[2] Price D. Pentium FDIV flaw-lessons learned // Micro, IEEE. 1995. Vol. 15. No 2. P. 86.

[3] ГОСТ Р ИСО 9000-2001. Государственный стандарт Российской Федерации. Система менеджмента качества. Основные положения и словарь. ИПК Издательство стандартов, 2001. С. 19.

[4] Карпов Ю.Г. Model Checking. Верификация параллельных и распределенных программных систем. СПб.: БХВ-Петербург, 2010. 560 с.

[5] Clarke E., Grumberg O., Peled D. Model checking. The MIT Press, 1999. 273 p.

[6] Chris Spear SystemVerilog for Verification, Springer. 2006. 536 p.

[7] Kamkin A. Coverage-Directed Verification of Microprocessor Units Based on Cycle-Accurate Contract Specifications // Proceedings of EWDTS 2008: The 6th East-West Design & Test Symposium. 2008. P. 84–87.

[8] Камкин А.С. Метод автоматизации имитационного тестирования микропроцессоров с конвейерной архитектурой на основе формальных спецификаций: автореф. дис. : к-та. тех. наук. А.С. Камкина ; М., 2009. 21 с.

[9] Белкин В. В., Шаршунов С. Г. Разработка функциональных тестов конвейеризованных процессоров на основе высокоуровневых моделей // Приборы и системы. Управление, контроль, диагностика. 2007. №4. С. 22-27.

[10] Ho R. Validation Tools for Complex Digital Designs. PhD Thesis. 1996. 113 p.