На правах рукописи

Буренков Владимир Сергеевич

МЕТОДЫ И СРЕДСТВА ВЕРИФИКАЦИИ ПРОТОКОЛОВ КОГЕРЕНТНОСТИ ПАМЯТИ

Специальность 05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

Автореферат
диссертации на соискание ученой степени
кандидата технических наук

Москва – 2017

Работа выполнена в федеральном государственном бюджетном образовательном учреждении высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана) на кафедре компьютерных систем и сетей.

Научный руководитель: **Иванов Сергей Ростиславович**,
кандидат технических наук, доцент кафедры компьютерных систем и сетей МГТУ им. Н.Э. Баумана

Официальные оппоненты: **Лацис Алексей Оттович**,
доктор физико-математических наук, заведующий сектором Федерального государственного учреждения «Федеральный исследовательский центр Институт прикладной математики им. М.В. Келдыша Российской академии наук»

Чибисов Петр Александрович,
кандидат технических наук, заведующий сектором Федерального государственного учреждения «Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук»

Ведущая организация: Акционерное общество «Научно-исследовательский центр электронной вычислительной техники»

Защита состоится «25» мая 2017 г. в 11 часов на заседании диссертационного совета Д 002.087.01 при Институте системного программирования РАН по адресу: 109004, Москва, ул. А. Солженицына, д. 25.
С диссертацией можно ознакомиться в библиотеке и на сайте Федерального государственного бюджетного учреждения науки Институт системного программирования Российской академии наук.

Автореферат разослан «____» _____ 2017 г.

Ученый секретарь
диссертационного совета Д 002.087.01,
кандидат физико-математических наук **Зеленов С. В.**

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы исследования.

Мультипроцессоры с общей памятью составляют один из базовых классов высокопроизводительных вычислительных систем. В последнее время системы, относящиеся к данному классу, получили развитие в форме многоядерных микропроцессоров, объединяющих несколько процессоров (ядер) на одном кристалле. Характерно, что количество ядер таких микропроцессоров постоянно увеличивается. Разработкой многоядерных микропроцессоров и мультипроцессорных комплексов занимаются как зарубежные (в частности, IBM, Intel и AMD), так и российские компании (в частности, АО «МЦСТ» и ПАО «ИНЭУМ им. И. С. Брука»).

Основной проблемой, возникающей при создании мультипроцессоров с общей памятью, является обеспечение согласованного (когерентного) состояния памяти. Каждое ядро имеет в своем составе локальную кэш-память, из-за чего в системе могут сосуществовать несколько копий одних и тех же данных: одна копия в основной памяти и несколько копий в кэш-памяти процессоров. При изменении какой-либо копии другие копии должны быть либо удалены, либо изменены согласованным образом. За это отвечают так называемые кэш-контроллеры – устройства подсистемы памяти, объединенные в сеть и взаимодействующие друг с другом по специальному протоколу – *протоколу когерентности*.

Разработка механизмов, отвечающих за когерентность памяти, осуществляется в два этапа: проектирование протокола когерентности и реализация протокола в аппаратуре. Ввиду сложности современных протоколов на обоих этапах возможны ошибки. Ошибки в протоколе когерентности особенно критичны и должны быть выявлены до начала реализации подсистемы памяти. Допущенные ошибки в протоколе могут привести к ошибкам в реализации, которые не будут найдены до выпуска микропроцессора

и проявят себя только во время реальной работы микропроцессора.

Актуальность разработки методов верификации протоколов когерентности памяти отмечается многими учеными, в число которых входят Э. М. Кларк, Э. А. Эмерсон, А. Пнуэли, Д. Пелед, Л. Лэмпорт, С. Граф, К. МакМиллан, О. Грамберг, П. А. Абдулла, С. Парк, Д. Л. Дилл, М. Талупур, И. В. Коннов и др. В их работах большое внимание уделяется разработке формальных методов, которые позволяют получить математическое доказательство соответствия модели верифицируемого протокола когерентности его спецификации, то есть набору свойств, которым он должен удовлетворять. Несмотря на большое количество исследований в данной области, вопросу верификации протоколов когерентности памяти продолжает уделяться повышенное внимание.

К сожалению, наиболее известные формальные методы имеют ограниченное применение при верификации протоколов когерентности памяти. Полностью автоматизируемый метод проверки моделей связан с исследованием очень большого числа состояний и поэтому на практике позволяет верифицировать только протоколы систем с четырьмя (или менее) ядрами. Масштабируемые методы дедуктивной верификации требуют чрезмерного объема ручной работы.

В связи с потребностью в верификации систем с большим количеством ядер, в данной работе поставлена задача параметризованной верификации протоколов когерентности памяти, предполагающая исследование систем с любым числом ядер. По результатам выполненного в диссертации аналитического обзора работ, в которых представлены методы параметризованной верификации протоколов когерентности памяти, проведена классификация таких методов. Установлено, что существующие методы, как правило, либо требуют значительного объема ручной работы, либо, в случае возможности их полной автоматизации, накладывают большое количество ограничений на модели протоколов, что препятствует верификации протоколов

когерентности памяти современных микропроцессоров. Инструментальная поддержка существующих методов зачастую выполнена в виде экспериментальных прототипов, которые не всегда находятся в открытом доступе. Данные обстоятельства определяют чрезвычайную актуальность разработки новых методов и средств верификации протоколов когерентности памяти.

Цели и задачи работы.

Цель работы – разработка методов и средств верификации протоколов когерентности памяти масштабируемых микропроцессорных систем, разрабатываемых для серийного выпуска.

Для достижения цели работы были поставлены следующие **задачи**:

1. Провести анализ существующих методов верификации протоколов когерентности памяти.
2. Определить совокупность математических объектов, необходимых для представления протоколов когерентности памяти и разработать модель протоколов когерентности памяти.
3. Разработать метод верификации протоколов когерентности памяти, который являлся бы масштабируемым и обеспечивал высокий уровень автоматизации процесса проверки.
4. Доказать корректность разработанного метода с использованием математической модели протоколов когерентности памяти.
5. Разработать программный инструмент, позволяющий осуществлять верификацию протоколов когерентности с использованием разработанного метода.
6. Провести экспериментальные исследования предложенного метода в применении к микропроцессору Эльбрус-4С.

Научная новизна.

Научной новизной обладают следующие результаты работы:

1. Разработанный автором метод построения формальных моделей

протоколов когерентности памяти, основанный на синтаксических преобразованиях Promela-моделей, позволяющий существенно сократить пространство исследуемых состояний.

2. Сформулированная и доказанная теорема о сохранении разработанными преобразованиями свойств, которым должен удовлетворять верифицируемый протокол когерентности памяти (свойств-инвариантов). Данная теорема определяет корректность предложенного метода верификации.

3. Разработанные алгоритмы преобразования дерева абстрактного синтаксиса языка Promela, которое является промежуточным представлением Promela-моделей, позволяющие автоматически выполнять предложенные преобразования моделей.

Теоретическая значимость работы заключается в том, что:

1. Разработана математическая модель протоколов когерентности памяти. Данная модель позволяет установить связь между синтаксическими преобразованиями текстов моделей, написанных на языке Promela, и результирующими преобразованиями соответствующих структур, исследуемых инструментальным средством Spin в ходе верификации.

2. Сформулирована и доказана теорема о сохранении разработанными преобразованиями свойств-инвариантов.

3. Доказана корректность предложенной процедуры уточнения формальных моделей, которая используется для устранения ложных сообщений об ошибках.

Практическая значимость работы:

1. Предложен подход к разработке Promela-процессов, моделирующих кэш-контроллеры, и Promela-процессов, представляющих системный коммутатор, который координирует работу кэш-контроллеров. Подход предоставляет возможность представления моделей протоколов когерентности памяти в виде, используемом разработчиками протоколов, то есть в виде множества взаимодействующих конечных автоматов.

2. Разработан программный инструмент, позволяющий с помощью операций над деревом абстрактного синтаксиса, которое является промежуточным представлением исходной модели протокола когерентности памяти, автоматизировать преобразования Promela-моделей.

3. Разработана методика параметризованной верификации формальных моделей протоколов когерентности памяти.

Разработанные методы и средства реализованы в программной системе, с помощью которой проведена верификация протокола когерентности 16-ядерной системы из микропроцессоров Эльбрус-4С, разработанной в АО «МЦСТ». Эти результаты могут также найти применение в разработке других современных многоядерных микропроцессоров.

Методология и методы исследования.

Результаты диссертационной работы получены на базе использования методов и моделей, применяемых для проведения формальной верификации. Применен формализованный подход, предполагающий использование математической модели для представления верифицируемого объекта. Математическую основу составляют теория множеств, теория графов, теория алгоритмов, математическая логика, теория формальных языков и теория автоматов.

Положения, выносимые на защиту.

1. Метод верификации протоколов когерентности памяти современных мультипроцессорных систем, основанный на разработанных синтаксических преобразованиях моделей, написанных на языке Promela, и разработанной процедуре уточнения моделей, получаемых в результате применения преобразований.

2. Программный инструмент, автоматизирующий процесс преобразования Promela-моделей к виду, позволяющему проводить верификацию при помощи инструмента Spin.

Достоверность и апробация результатов.

Полученные в работе научные положения имеют математическое обоснование:

- разработана модель в виде канальной системы, позволяющая формализовать семантику Promela-моделей протоколов когерентности памяти;
- сформулирована и доказана теорема, определяющая корректность предложенного метода верификации.

Основные положения работы обсуждались на:

- VII Всероссийской молодежной научно-инженерной выставке «Политехника» в МГТУ им. Н.Э. Баумана (г. Москва, 2012 г.);
- 9-м международном коллоквиуме молодых ученых по программной инженерии SYRCoSE (г. Самара, 2015 г.);
- 10-м международном коллоквиуме молодых ученых по программной инженерии SYRCoSE (г. Москва, 2016 г.);
- 7-й Всероссийской научно-технической конференции «Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС-2016)» (г. Москва, 2016 г.);
- 14-м международном научно-технологическом симпозиуме IEEE East-West Design and Test (EWDTS-2016) (г. Ереван, 2016 г.);
- Всероссийском конкурсе научно-исследовательских работ в области инженерных и гуманитарных наук, проводимом в рамках Всероссийского инновационного молодежного научно-инженерного форума «Политехника». По итогам конкурса автор диссертации объявлен победителем и награжден дипломом первой степени.

Все основные теоретические и практические результаты работы: метод, алгоритмы, методика и программные средства внедрены в проектирование микропроцессоров. Они использованы при выполнении опытно-конструкторских работ по темам «Экскурсовод-2», «Процессор-1» и

«Процессор-9» в АО «МЦСТ», о чем имеется акт о внедрении.

Публикации. По теме диссертации опубликовано 16 научных работ, в том числе 8 научных статей [1–8] в рецензируемых журналах, входящих в перечень журналов, рекомендованных ВАК РФ. В работе [9] автором диссертации описан подход к верификации с помощью метода проверки моделей. В работах [6, 15] автором диссертации обозначены и рассмотрены проблемы верификации протоколов когерентности памяти и пути их решения применительно к протоколу Эльбрус-4С. В работах [4, 5, 7] представлен метод верификации, разработанный автором диссертации. В работе [16] представлены результаты экспериментов, проведенных автором диссертации. Архитектура разработанного программного инструмента описана в работе [8].

Личный вклад автора. Все представленные в диссертации результаты получены лично автором.

Объем и структура работы. Диссертация состоит из введения, четырех глав, заключения и списка литературы, занимающих 161 страницу. Список литературы включает 122 наименования.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении обоснована актуальность темы диссертации. Сформулированы цель и основные задачи исследования, решение которых необходимо для достижения поставленной цели. Дана общая характеристика работы.

В первой главе проведен анализ существующих методов и средств верификации протоколов когерентности памяти. Проведена классификация методов параметризованной верификации протоколов когерентности памяти.

За основу предлагаемого метода взяты работы Чау, Маннавы, Парка, Таттла и Талупура, в которых изложен метод, основанный на синтаксической абстракции моделей, написанных на языке Murφ, и способный к масштабированию. В приведенных работах отсутствует определение

ограничений на модели протоколов, упоминаемые авторами данных работ инструментальные средства по преобразованию моделей не находятся в открытом доступе, что делает невозможным применение метода без его доработки и без создания реализующих его программных инструментов. Сформулирована постановка задачи.

Во второй главе разработаны математическая модель протоколов когерентности памяти и метод построения формальных моделей, основанный на синтаксических преобразованиях Promela-моделей. Сформулирована и доказана теорема о сохранении синтаксическими преобразованиями свойств, которым должен удовлетворять верифицируемый протокол когерентности памяти, определяющая корректность предложенного метода.

В работе выбрана модель протоколов когерентности памяти в виде множества взаимодействующих конечных автоматов, элементы которого представляют два типа устройств: кэш-контроллеры и системный коммутатор (рисунок 1). Кэш-контроллеры управляют кэш-памятью, а их работу координирует системный коммутатор.

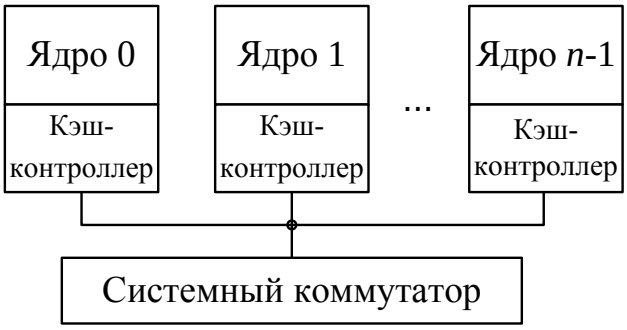

Рисунок 1 – Структура микропроцессора Эльбрус-4С

Такая модель соответствует структуре микропроцессора Эльбрус-4С. В данном микропроцессоре каждое ядро обладает собственной кэш-памятью первого и второго уровней. Кэш-контроллер управляет кэш-памятью только второго уровня, поэтому в данной работе рассматривается верификация протоколов когерентности памяти, в которых за когерентность отвечает один уровень кэш-памяти.

На основании документации конечные автоматы – кэш-контроллер и системный коммутатор – представляются в виде таблиц или диаграмм. Кэш-контроллеры имеют в качестве основных состояния из широко распространенного множества Modified, Owned, Exclusive, Shared, Invalid. Переходы между данными состояниями не являются атомарными, а происходят через промежуточные состояния. Промежуточные состояния являются специфичными для каждого микропроцессора и являются одним из факторов, определяющих высокий уровень сложности верификации.

Условия, определяющие корректность протокола когерентности, формулируются относительно основных состояний кэш-контроллеров:

1. Кэш-строка никогда не может находиться в состоянии Modified в двух кэшах одновременно.
2. Кэш-строка никогда не может находиться в состоянии Owned в двух кэшах одновременно.
3. Кэш-строка никогда не может находиться в одном кэше в состоянии Modified, а в другом – в состоянии Shared или Owned.

Для определения способа представления модели в виде множества взаимодействующих конечных автоматов с помощью ЭВМ был проведен анализ существующих систем верификации Murφ, NuSMV и Spin, основанных на методе проверки моделей, и языков описания моделей, предоставляемых этими инструментами. На основе анализа выбран язык описания моделей Promela, поскольку он, в отличие от других языков, предлагает абстракции, позволяющие представить множество взаимодействующих конечных автоматов. Конечные автоматы в данном языке представляются процессами, а их асинхронное взаимодействие моделируется посредством каналов (FIFO-очередей). Поэтому математическая модель протоколов когерентности, используемая в данной работе, основана на формальной семантике Promela-моделей.

Математическая модель состоит из нескольких уровней. Promela-

процессам соответствуют графы процессов. Асинхронная композиция графов процессов описывается канальной системой. Путем «развертывания» канальной системы получается система переходов – стандартная модель аппаратных и программных систем.

В качестве языка спецификации протоколов выбрана темпоральная логика линейного времени, которая, согласно своей семантике, позволяет формализовать приведенные выше условия корректности. Эти условия представляют собой свойства-инварианты:

1. $\forall i, j \in \{1, \dots, n\}, i \neq j$: $\mathrm{G}\neg(cache[i] = M \wedge cache[j] = M)$.
2. $\forall i, j \in \{1, \dots, n\}, i \neq j$: $\mathrm{G}\neg(cache[i] = O \wedge cache[j] = O)$.
3. $\forall i, j \in \{1, \dots, n\}, i \neq j$: $\mathrm{G}\neg\big(cache[i] = M \wedge (cache[j] = O \vee cache[j] = S)\big)$.

Системой переходов TS называется шестерка $TS = (S, Act, \rightarrow, I, AP, L)$, где S – множество состояний, Act – множество действий, $\rightarrow \subseteq S \times Act \times S$ – отношение переходов, $I \subseteq S$ – множество начальных состояний, AP – множество атомарных высказываний, $L: S \rightarrow 2^{AP}$ – функция пометок.

Поведение процессов описывается с помощью операторов языка Promela. Обозначим множество всех переменных модели через Var, множество каналов модели – $Chan$. Обозначим через $Cond(Var)$ множество логических выражений относительно переменных $v \in Var$, а через $Cond(Var, Chan)$ множество логических выражений относительно переменных $v \in Var$ и каналов $c \in Chan$. Формальная семантика оператора языка Promela с переменными из множества Var и каналами из множества $Chan$ представляется графом процесса над $(Var, Chan)$ – ориентированным графом, ребра которого помечены условиями над элементами $(v, c) \in Var \times Chan$ и действиями.

Множество действий представлено объединением $Act \cup Comm$, где элементы множества Act определяются четырьмя базовыми операторами языка Promela: присваивание, оператор assert, оператор print, выражение, а элементы

множества $Comm$ – двумя операторами: отправка сообщения в канал и извлечение сообщения из канала.

Графом процесса PG, или процессом, над $(Var, Chan)$ называется шестерка $PG = (Loc, Act, Effect, \hookrightarrow, Loc_0, g_0)$, где Loc – множество состояний (вершин) графа, Act – множество действий, $Effect: Act \times Eval(Var) \to Eval(Var)$ – функция, определяющая результат действий, $\hookrightarrow \in Loc \times Cond(Var, Chan) \times (Act \cup Comm) \times Loc$ – отношение переходов, $Loc_0 \subseteq Loc$ – множество начальных состояний, $g_0 \in Cond(Var, Chan)$ – начальное условие.

Канальной системой CS над $(Var, Chan)$ называется множество графов процессов PG_i над $(Var_i, Chan)$, где $1 \leq i \leq n$ и $Var = \bigcup_{1 \leq i \leq n} Var_i$. Обозначение: $CS = [PG_1 \mid \ldots \mid PG_n]$.

Семантика канальной системы формализуется посредством системы переходов, как показано в тексте диссертации.

В качестве математической модели протокола когерентности в работе использована канальная система $CS = [PG_0 \mid PG_1 \mid \ldots \mid PG_n]$, где PG_0 – граф процесса, соответствующий системному коммутатору home-процессора, $PG_1, \ldots, PG_n$ – идентичные графы процессов, соответствующие контроллерам кэш-строки, находящимся в кэшах верифицируемой системы. Среди переменных из Var выделено множество подмножеств $\{Vstate_i \subseteq Var \mid i = 0, \ldots, n\}$ переменных, описывающих состояние процессов $PG_i, 0 \leq i \leq n$. Также введено множество переменных V_{loca}, элементы которого находятся во взаимно-однозначном соответствии с множеством локальных переменных того из процессов $PG_i, 2 < i \leq n$, который использует эти локальные переменные для модификации переменных из $\bigcup_{0 \leq i \leq 2} Vstate_i$.

Проверяемые свойства протокола затрагивают не более двух кэшей. Поскольку все кэш-контроллеры идентичны и взаимозаменяемы, не имеет значения, какие именно два индекса графов $PG_1, \ldots, PG_n$ рассматривать. Поэтому без потери общности будем считать, что рассматриваются графы PG_1 и

PG_2, и все проверяемые свойства сформулированы только относительно части системы, определяемой этими графами.

В отсутствие свойств относительно $PG_3, \dots, PG_n$, с точки зрения графов PG_0, PG_1, PG_2, конкретное значение индекса графов $PG_3, \dots, PG_n$ неважно. Поэтому выполняется следующее консервативное преобразование системы CS.

Предполагается, что в исходной системе среди переменных $v \in Vstate_i, 0 \le i \le n$, хранящих информацию о других процессах, могут быть только переменные, хранящие номер процесса, то есть переменные, областью определения которых является множество $\{0, \dots, n\}$. В случае, когда полностью рассматривается только состояние процессов $PG_i, 0 \le i \le 2$, важно сохранять точные значения лишь для случаев, когда это значение принадлежит множеству $\{0,1,2\}$; все остальные значения неразличимы, и их можно представить некоторым абстрактным значением, не находящимся в этом множестве. Поэтому множества значений переменных $v \in Var$ и каналов $c \in Chan$, таких, что $dom(v) = \{0, \dots, n\}$ и $dom(c.id) = \{0, \dots, n\}$ соответственно, заменяются на $dom_{abs}(v) = \{0,1,2, ABS\}$ и $dom_{abs}(c) = \{0,1,2, ABS\}$, соответственно, где $ABS > 2$ – некоторая константа. Для всех остальных случаев $dom_{abs}(v) = dom(v), dom_{abs}(c) = dom(c)$. Далее под исходной системой будем понимать преобразованную таким образом систему CS.

В соответствии с предлагаемым методом, после проведения абстракции типов данных осуществляется замена исходной модели $CS = [PG_0 \mid PG_1 \mid \dots \mid PG_n]$ на модель $CS_{abs} = [PG_0 \mid PG_1 \mid PG_2 \mid PG_3]$, в которой изначально графы $PG_i, 0 \le i \le 3$ являются графами $PG_i, 0 \le i \le 3$ исходной системы CS, соответственно (рисунок 2).

Поскольку абстрактная модель должна полностью представлять состояние процессов PG_0 (координатора), PG_1, PG_2 (двух кэш-контроллеров) и состояние $PG_3, \dots, PG_n$, используемое при модификации состояния процессов PG_0, PG_1, PG_2, то в качестве пометок состояний абстрактной TS_{abs} и исходной

TS систем выступает следующее множество: $AP = Cond(V_{AP})$, где $V_{AP} = \bigcup_{0 \le i \le 2} Vstate_i \cup V_{loca}$.

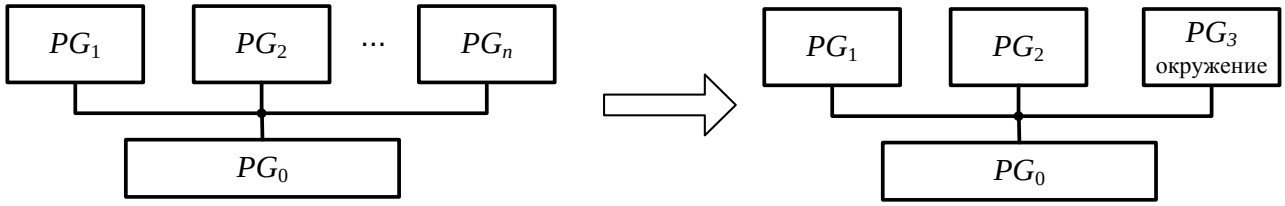

Рисунок 2 – Трансформация канальной системы при абстрактных преобразованиях

Последующая модификация графов $PG_i, 0 \le i \le 3$ осуществляется посредством синтаксических преобразований операторов, из которых состоят Promela-модели.

Преобразования операторов присваивания приведены в таблице 1. Других операторов присваивания, кроме приведенных, нет. Символ ∅ означает удаление рассматриваемого оператора.

Таблица 1 – Преобразование присваиваний

Оператор в исходной модели	Оператор в абстрактной модели
$v = val\ (v \in Var, val \in dom(v))$	$v = val$, если $v \in \bigcup_{i=0,1,2} Vstate_i$, ∅, если $v \in \bigcup_{2<i \le n} Vstate_i$

Преобразования выражений $(v, c) \in Cond(V, C)$ осуществляются следующим образом. Если $V \subseteq \bigcup_{i=0,1,2} Vstate_i \cup V_{loca}$, то v остается неизменным, иначе v заменяется на истину. Если $C \subseteq C_1 \cup \{c_{2,1}, c_{2,2}\} \cup C_3$, то c остается неизменным, иначе c заменяется на истину.

Исследование вариантов разработанных моделей протоколов когерентности памяти на примере протокола системы Эльбрус-4С показало, что множество каналов модели представимо тремя подмножествами:

• C_1 – множество каналов емкости n, в которые могут отправлять сообщения процессы $PG_i, 1 \le i \le n$. Извлекать сообщения может только один

процесс $PG_i, 0 \le i \le n$ на определенном этапе выполнения запроса.

- $C_2 = \{c_{2,1}, \dots, c_{2,n}\}$ – множество каналов емкости $m \in \mathbb{N}$, таких, что из канала $c_{2,i}, 1 \le i \le n$ может извлекать сообщения только PG_i, а отправлять сообщения по этому каналу может только PG_0.

- C_3 – множество каналов, по которым может отправлять сообщения только один процесс на определенных этапах выполнения запроса. Для каналов этого множества характерно соответствие каждому оператору приема сообщения ровно одного оператора отправки сообщения.

Такое разбиение множества каналов позволило на основе особенностей операторов, представляющих коммуникационные действия, разработать различные преобразования таких операторов, необходимые для получения абстрактных моделей. Данные преобразования представлены в таблицах 2–4.

Сообщения, передаваемые по каналам, являются парами $m = \langle opc, id \rangle$, где $opc \in \mathbb{N}$ не является номером процесса, $id \in \{0, \dots, n\}$ является номером процесса (например, идентификатором отправителя сообщения).

Таблица 2 – Преобразование коммуникационных действий для $c \in C_1$

Оператор в исходной модели	Оператор в абстрактной модели
$c!\, v\ (v \in dom(c))$	$c!\, v$, если оператор находится в PG_1, PG_2 $\emptyset$, если оператор находится в PG_3
$c?\, x\ (x \in Var, dom(x) \supseteq dom(c))$	Недетерминированный выбор

Замена оператора atomic { guard_abs –> c?x; } оператором недетерминированного выбора

```
if
:: atomic { guard_abs –> c?x; }
:: atomic { m.opc = opc1; m.id = ABS; }
…
:: atomic { m.opc = opck; m.id = ABS; }
fi;
```

осуществляется так, что множество $\{opc_1, \dots, opc_k\} \subseteq dom_{abs}(m.opc)$ содержит

в себе все возможные значения, которые в исходной модели могли быть отправлены с помощью соответствующих операторов отправки сообщения, находящихся в $PG_3, \dots, PG_n$. Здесь guard_abs получается согласно преобразованиям выражений.

Таблица 3 – Преобразование коммуникационных действий для $c \in C_2$

Оператор в исходной модели	Оператор в абстрактной модели
$c!\,v\ (v \in dom(c))$	$c!\,v$, если $c = c_{2,1}$ или $c = c_{2,2}$ $\emptyset$, если $c \in C_2 \backslash \{c_{2,1}, c_{2,2}\}$
$c?\,x\ (x \in Var, dom(x) \supseteq dom(c))$	$c?\,x$, если $c = c_{2,1}$ или $c = c_{2,2}$ $\emptyset$, если $c \in C_2 \backslash \{c_{2,1}, c_{2,2}\}$

Таблица 4 – Преобразование коммуникационных действий для $c \in C_3$

Оператор в исходной модели	Оператор в абстрактной модели
$c!\,v\ (v \in dom(c))$	$c!\,v$
$c?\,x\ (x \in Var, dom(x) \supseteq dom(c))$	$c?\,x$

Корректность предложенных преобразований определяется доказанной в диссертации теоремой о сохранении преобразованиями свойств-инвариантов. В следующей теореме доказано, что множество достижимых состояний системы $TS_{abs} = (S_{abs}, Act_{abs}, \rightarrow_{abs}, I_{abs}, AP, L_{abs})$ включает в себя множество достижимых состояний системы $TS = (S, Act, \rightarrow, I, AP, L)$.

Теорема. Пусть p – некоторая формула пропозициональной логики, составленная относительно атомарных высказываний из множества AP. Для всех состояний $s \in S$, таких, что s достижимо из некоторого начального состояния $s_0 \in I$ и $s \vDash p$, существует состояние $f(s) \in S_{abs}$, достижимое из некоторого начального состояния $f(s_0) \in I_{abs}$, такое, что $f(s) \vDash p$.

Теорема доказана методом математической индукции по длине пути системы переходов TS.

Соответствие переходов в исходной и абстрактной системах переходов

показано на рисунках 3 – 6. Как видно, в абстрактной системе переходов нескольким состояниям исходной системы переходов может соответствовать одно, только если у этих состояний исходной системы одинаковая пометка.

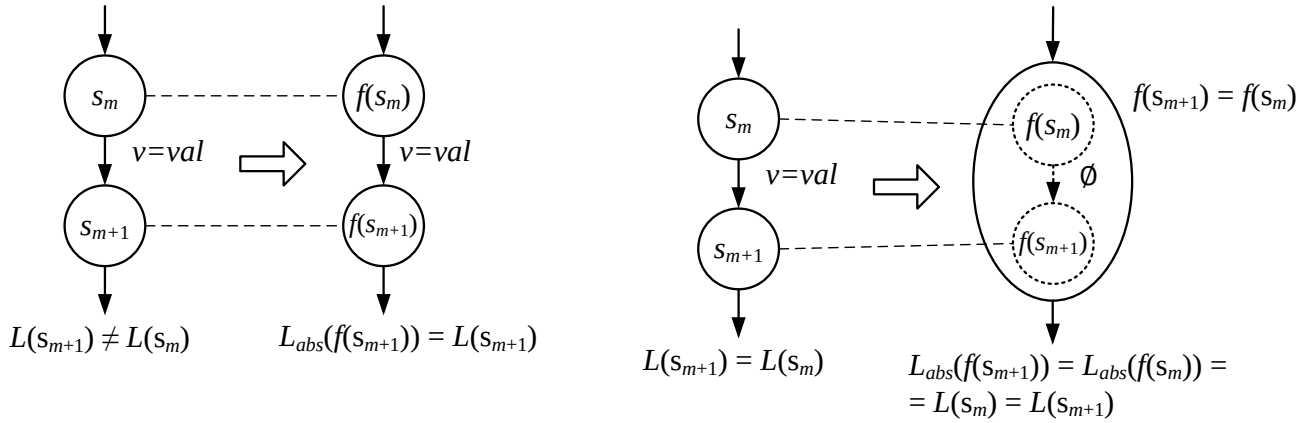

Рисунок 3 – Соответствие перехода между двумя состояниями с разными пометками в исходной системе переходу в абстрактной системе при преобразовании присваиваний (слева) и отображение двух состояний с одинаковыми пометками в исходной системе в одно состояние в абстрактной системе при преобразовании присваиваний (справа)

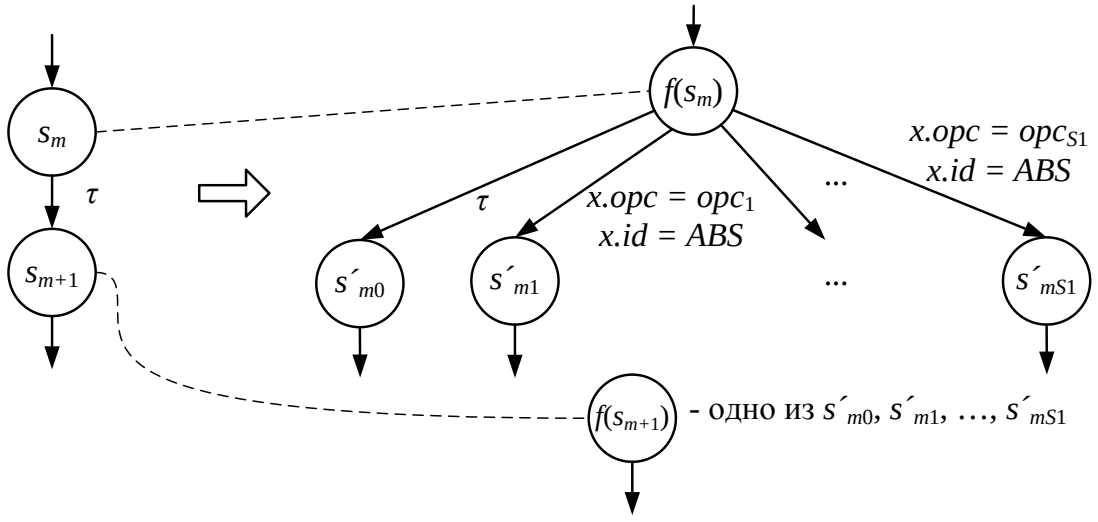

Рисунок 4 – Соответствие перехода в исходной системе, порожденного оператором считывания из канала $c \in C_1$, множеству переходов в абстрактной системе

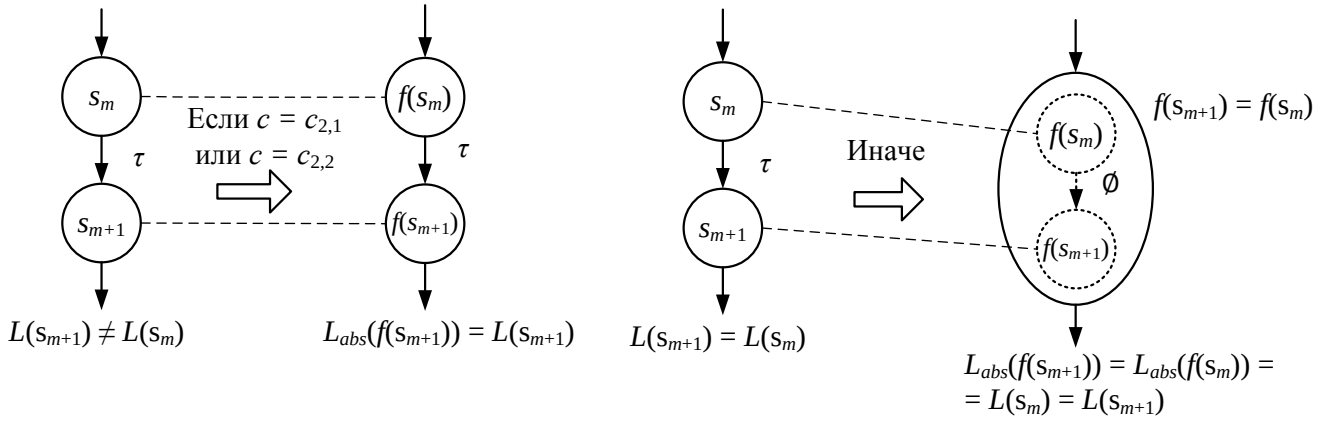

Рисунок 5 – Соответствие перехода в исходной системе, порожденного оператором считывания из канала $c \in C_2$, переходу в абстрактной системе

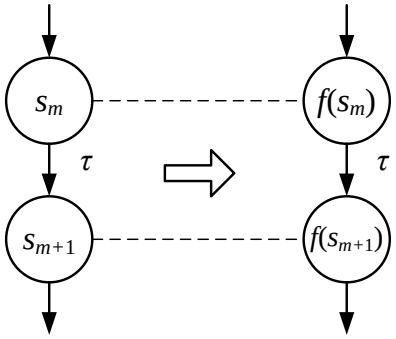

Рисунок 6 – Соответствие перехода в исходной системе, порожденного оператором считывания из канала $c \in C_3$, переходу в абстрактной системе

В третьей главе разработана процедура уточнения абстрактных моделей, получаемых с помощью метода, предложенного в главе 2. Детально определено представление ряда множеств, введенных в главе 2, в Promela-моделях. Сформулированы преобразования текста Promela-моделей, в соответствии с более абстрактным описанием преобразований в главе 2.

Для уточнения абстрактных моделей, выполняемого с целью устранения ложных контрпримеров, предложена следующая процедура:

1. Для каждого типа исходных запросов определить на основе документации частично упорядоченное множество $(A, \prec)$ действий ($\prec$ – отношение строгого порядка):

$\forall a_1, a_2 \in A: a_1 \prec a_2$, если действие a_1 осуществляется раньше

действия a_2.

2. Пока имеются ложные контрпримеры:

2.1. Найти действие a, которое привело к возникновению контрпримера. Найти множество A, содержащее действие a: $a \in A$. В множестве A найти действие b, такое что $b \prec a$.

2.2. Ввести переменную логического типа aux_b с начальным значением $false$. Заменить b в модели на атомарную последовательность $b; aux_b := true$.

3. Добавить aux_b к защите команды, содержащей действие a, с помощью логического И. Заменить a на атомарную последовательность $a; aux_b := false$.

В тексте диссертации приведено доказательство корректности данной процедуры. Получаемая модель сохраняет свойства-инварианты.

Разработанный в диссертации метод параметризованной верификации протоколов когерентности памяти основан на:

1. Применении разработанных абстрактных преобразований к Promela-моделям протоколов когерентности памяти, представляющих протоколы когерентности в виде множества процессов, обменивающихся сообщениями посредством каналов.

2. Уточнении абстрактных моделей на основе анализа контрпримеров и введения вспомогательных переменных согласно разработанной процедуре уточнения.

Предложенная методика верификации протоколов когерентности памяти состоит из следующих этапов (рисунок 7):

1. Разработка исходной модели протокола когерентности на языке Promela. Модель разрабатывается верификатором вручную. На основании документации на протокол когерентности памяти, верификатор в соответствии с предложенным подходом составляет описание процессов, моделирующих кэш-контроллеры и системный коммутатор, и добавляет в модель необходимые

инфраструктурные элементы (определение каналов, запуск процессов).

2. Построение абстрактной модели в соответствии с предложенными синтаксическими преобразованиями Promela-моделей. Преобразования могут быть выполнены вручную либо автоматически с использованием системы построения абстрактной модели, разработка которой описана в главе 4.

3. Проверка выполнения свойств на абстрактной модели с помощью инструментального средства Spin. Данный этап идентичен проверке любых Promela-моделей.

4. Анализ верификатором отчета о верификации, сгенерированного инструментом Spin. Если отчет показывает отсутствие ошибок, то верификация завершается с вердиктом о корректности протокола когерентности памяти. Если отчет сигнализирует о наличии ошибки, то проводится анализ соответствующего контрпримера, сгенерированного системой Spin. Если верификатор устанавливает, что данный контрпример является ложным, то есть представленная в нем последовательность действий невозможна в реальной микропроцессорной системе, то производится уточнение абстрактной модели в соответствии с разработанной процедурой и переход к шагу 3. Если верификатор устанавливает, что данный контрпример представляет ошибку в протоколе когерентности, то о данной ошибке сообщается разработчикам протокола. После исправления ошибки разработчиками протокола, верификатор вносит соответствующие исправления в исходную Promela-модель протокола, и верификация начинается заново (происходит переход к шагу 1).

Последовательность указанных этапов повторяется до тех пор, пока не будут устранены все контрпримеры.

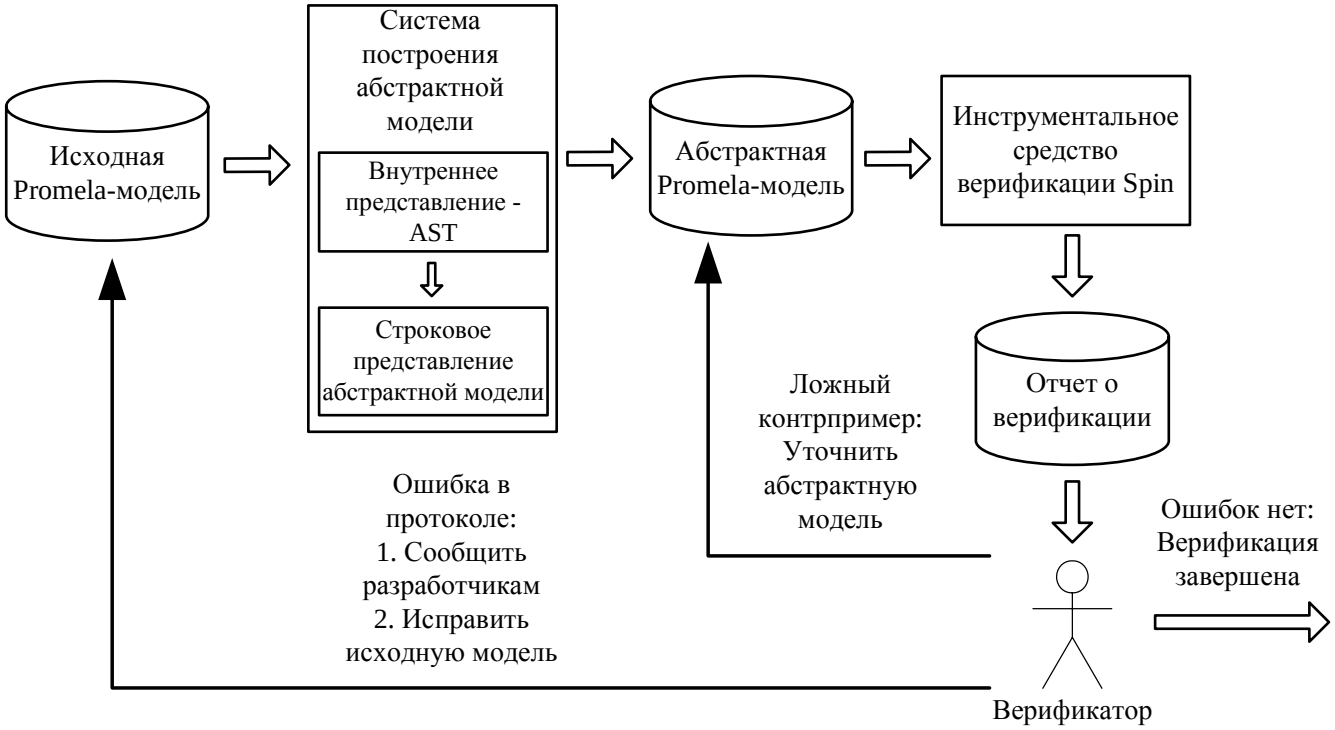

Рисунок 7 – Схема процесса верификации протоколов когерентности с памяти с использованием разработанной методики

В четвертой главе представлен процесс составления формальных моделей протоколов когерентности на примере протокола когерентности системы на кристалле Эльбрус-4С. Рассмотрен процесс уточнения абстрактных моделей протокола когерентности Эльбрус-4С.

Описана разработка инструментального средства, позволяющего автоматизировать преобразования Promela-моделей, приводящие к получению абстрактных моделей. Осуществлен выбор инструмента, позволяющего сконструировать дерево абстрактного синтаксиса на основе грамматики языка Promela и ряда структур данных, – библиотеки Boost.Spirit. Разработаны алгоритмы, позволяющие на этапе обхода дерева абстрактного синтаксиса выполнить все преобразования Promela-моделей, введенные в главе 3. Осуществлен сбор и обработка информации по результатам верификации протокола когерентности Эльбрус-4С.

В ходе процесса анализа документации и написания формальных моделей была обнаружена серьезная ошибка в протоколе когерентности. Также была

проведена верификация нескольких ошибочных версий протокола, разработанных на основе анализа ошибок, найденных в микропроцессорах Эльбрус. Во всех случаях ошибки были зафиксированы.

В таблицах 5–6 приведены количество состояний различных моделей и объем ресурсов, требуемых (после устранения всех контрпримеров) для проверки свойства $\mathrm{G}\neg(cache[1] = M \wedge cache[2] = M)$ при использовании оптимизации COLLAPSE, исключающей потери, реализованной в Spin. В модели отражены запросы, соответствующие обращениям к памяти с типами Write Back, Write Through и Write Combined. Именно для обращений с такими типами памяти в Эльбрус-4С поддерживается когерентность. Эксперименты проводились на сервере Intel Xeon E5-2697 (тактовая частота 2,6 ГГц) с 264 Гб оперативной памяти.

Таблица 5 – Требуемые ресурсы для верификации исходной модели

Количество ядер, представленных в модели	Число состояний модели	Объем оперативной памяти	Время верификации
3	$5{,}1 \cdot 10^6$	328 Мб	15 с
4	$1{,}3 \cdot 10^9$	81 Гб	1,5 часа

Таблица 6 – Требуемые ресурсы для верификации абстрактной модели

Количество ядер, представленных в модели	Число состояний модели	Объем оперативной памяти	Время верификации
любое, большее 2	$2{,}2 \cdot 10^6$	108 Мб	6,2 с

Таким образом, предложенный метод верификации Promela-моделей позволил сократить требование ресурсов (памяти) с чрезмерно большого (при количестве ядер, большем, чем 3–4), что свойственно методу проверки моделей, до незначительного. Объем ручной работы при верификации приемлем: для

уточнения полученной абстрактной модели потребовалось ввести две вспомогательные переменные логического типа.

ОСНОВНЫЕ ВЫВОДЫ И РЕЗУЛЬТАТЫ РАБОТЫ

1. Выполнен аналитический обзор методов параметризованной верификации протоколов когерентности памяти. Проведена классификация методов, указаны достоинства и недостатки как отдельных методов, так и групп методов.

2. Выбран язык Promela, позволяющий описывать протоколы когерентности памяти, представляя их в виде, используемом разработчиками протоколов, и разработана математическая модель протоколов когерентности памяти, основанная на формальной семантике языка Promela.

3. Разработан новый метод верификации протоколов когерентности памяти, базирующийся на преобразованиях Promela-моделей, приводящих к абстрактным моделям, и позволяющий существенно сократить число состояний модели протокола и проводить верификацию без ограничений на количество процессорных ядер. Поскольку все преобразования осуществляются на синтаксическом уровне, метод позволяет переиспользовать все множество алгоритмов верификации и оптимизаций, реализованное в инструменте Spin.

4. Сформулирована и доказана теорема, определяющая корректность предложенного метода, в которой проведен анализ соответствия переходов между состояниями в исходной и абстрактной моделях.

5. Сформулированы ограничения на модели протоколов когерентности и разработан новый подход к описанию моделей, базирующийся на предложенной автором диссертации организации Promela-процессов и использовании примитивов передачи сообщений.

6. Разработан программный инструмент, позволяющий с помощью операций над деревом абстрактного синтаксиса, которое является промежуточным представлением исходной модели, автоматизировать

преобразования Promela-моделей. Инструмент устраняет необходимость проведения большого объема ручной работы, при выполнении которой высока вероятность совершения серьезных ошибок, которые могут остаться незамеченными.

7. С использованием разработанных методов и средств проведена верификация протокола когерентности 16-ядерной системы из микропроцессоров Эльбрус-4С, разработанной в АО «МЦСТ», при незначительных затратах памяти и приемлемых временных затратах.

Направления дальнейших исследований по теме диссертации:

1. Разработка методов верификации иерархических протоколов когерентности памяти, в реализации которых задействованы несколько уровней кэш-памяти. Необходимость разработки таких методов обусловлена введением дополнительного уровня кэш-памяти (кэш-памяти третьего уровня), участвующего в реализации протокола когерентности, в новейших микропроцессорах архитектуры Эльбрус, проектируемых в настоящее время.

2. Разработка методов верификации аппаратных реализаций протоколов когерентности памяти. Сложность аппаратных реализаций значительно превышает сложность самих протоколов, и некоторые нетривиальные ошибки не удается найти на этапе верификации. Необходимы новые методы верификации, позволяющие повысить уверенность в корректности аппаратных реализаций. В этом направлении автором разработан программный инструмент, позволяющий получать тестовые программы на языке ассемблера Эльбрус на основе формальных моделей протоколов когерентности памяти. Данный инструмент использован для верификации реализаций протоколов когерентности памяти в микропроцессорах Эльбрус-4С, Эльбрус-8С и Эльбрус-8С2, проектируемых в настоящее время в АО «МЦСТ». Инструмент позволил найти более 60 ошибок, включая ошибки, найденные при проведении верификации на поздних стадиях проектирования и не обнаруженные ранее (в течение нескольких лет) другими методами.

СПИСОК РАБОТ, ОПУБЛИКОВАННЫХ ПО ТЕМЕ ДИССЕРТАЦИИ

Работы, опубликованные в изданиях, рекомендуемых ВАК:

1. Буренков, В. С. Анализ применимости инструмента Spin к верификации протоколов когерентности памяти / Буренков В. С. // Вопросы радиоэлектроники. – 2013. – Выпуск 3. – Сер. ЭВТ. – С. 126–134.

2. Буренков, В. С. Анализ применимости формальных методов к верификации протоколов когерентности кэш-памяти масштабируемых систем / Буренков В. С. // Вопросы радиоэлектроники. – 2015. – Выпуск 1. – Сер. ЭВТ. – С. 105–116.

3. Буренков, В. С. Генератор тестов для верификации протокола когерентности кэш-памяти / Буренков В. С. // Вопросы радиоэлектроники. – 2014. – Выпуск 3. – Сер. ЭВТ. – С. 56–63.

4. Буренков, В. С. Метод масштабируемой верификации PROMELA-моделей протоколов когерентности кэш-памяти / В. С. Буренков, А. С. Камкин // Сб. трудов VII Всероссийской научно-технической конференции «Проблемы разработки перспективных микро- и наноэлектронных систем – 2016». – 2016. – Часть II. – С. 54–60.

5. Буренков, В. С. О консервативном преобразовании формальных моделей, используемых применительно к масштабируемым системам для верификации протоколов когерентности памяти / Буренков В. С. // Вопросы радиоэлектроники. – 2016. – № 3. – Сер. ЭВТ. – С. 48–52.

6. Буренков, В. С. Проблемы формальной верификации технических систем / Буренков В. С., Иванов С. Р., Савельев А. Я. // Наука и образование: научное издание МГТУ им. Н.Э. Баумана. – 2012. – № 4.

7. Burenkov, V. Checking Parameterized PROMELA Models of Cache Coherence Protocols / Burenkov V. S., Kamkin A. S. // Proc. of the Institute for System Programming. – 2016. – Vol. 28, Issue 4. – P. 57–76.

8. Burenkov, V. On the Implementation of a Formal Method for Verification of Scalable Cache Coherent Systems / V. Burenkov // Proc. of the Institute for System Programming. – 2015. – Vol. 27, Issue 3. – P. 183–196.

Работы, опубликованные в других изданиях:

9. Буренков, В. С. Верификация технических систем методом проверки моделей / Буренков В. С., Иванов С. Р. // Современные компьютерные системы и технологии: сб. трудов каф. «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана. – М. : Изд-во НИИ Радиоэлектроники и лазерной техники, 2012. – С. 54–59.

10. Буренков, В. С. Генератор тестов для верификации протокола когерентности кэш-памяти [Электронный ресурс] / В. С. Буренков // Молодежный научно-технический вестник. – 2015. – № 2. – Режим доступа: http://sntbul.bmstu.ru/doc/759420.html.

11. Буренков, В. С. Инструмент верификации протокола когерентности памяти [Электронный ресурс] / В. С. Буренков // Молодежный научно-технический вестник. – 2013. – № 1. – Режим доступа: http://sntbul.bmstu.ru/doc/532989.html.

12. Буренков, В. С. Метод перебора состояний для верификации протоколов когерентности памяти / В. С. Буренков // Труды 54-й научной конференции МФТИ «Проблемы фундаментальных и прикладных естественных и технических наук в современном информационном обществе». – Москва–Долгопрудный–Жуковский : МФТИ, 2011. – Том 1. – С. 22–23.

13. Буренков, В. С. Метод проверки модели для верификации протоколов когерентности памяти / В. С. Буренков // Труды 55-й научной конференции МФТИ «Проблемы фундаментальных и прикладных естественных и технических наук в современном информационном обществе». – Москва–Долгопрудный–Жуковский : МФТИ, 2012. – Радиотехника и кибернетика : Том 1. – С. 60–61.

14. Буренков, В. С. Походы к верификации протоколов когерентности памяти [Электронный ресурс] / В. С. Буренков // Молодежный научно-технический вестник. – 2013. – № 8. – Режим доступа: http://sntbul.bmstu.ru/doc/603343.html.

15. Буренков, В. С. Проблемы параметризованной верификации протоколов когерентности памяти [Электронный ресурс] / Буренков В. С., Иванов С. Р. // Инженерный журнал: наука и инновации. – 2013. – № 11. – Режим доступа: http://www.engjournal.ru/catalog/it/hidden/1013.html.

16. Burenkov, V. Applying Parameterized Model Checking to Real-Life Cache Coherence Protocols / Vladimir Burenkov, Alexander Kamkin // Proc. of IEEE East-West Design & Test Symposium (EWDTS). – 2016. – P. 1–4.