УДК 004.25, 004.052.4

М.В. Петроченков, к.т.н. И.А. Стотланд (АО «МЦСТ»)

M. Petrochenkov, I. Stotland

МЕТОДИКА АВТОНОМНОЙ ВЕРИФИКАЦИИ УСТРОЙСТВ ПОДСИСТЕМЫ ПАМЯТИ МНОГОЯДЕРНЫХ МИКРОПРОЦЕССОРОВ

METHODS OF STANDALONE VERIFICATION OF MULTI-CORE MICROPROCESSOR MEMORY SUBSYSTEM UNITS

Приведена оценка классических подходов к автономной верификации устройств подсистемы памяти. Предложены методы проверки корректности, построенные на основе учета «подсказок» от реализации и применении утверждений. Описан подход к автономной верификации событийно-недетерменированных устройств, вводящий динамическое уточнение вариантов поведения модели на базе реакций от реализации. Приводится опыт применения рассмотренных методов при верификации устройств подсистемы памяти восьмиядерных микропроцессоров с архитектурой «Эльбрус».

The article presents the evaluation of the classical approaches to standalone verification of memory subsystem units. The methods based on using «prompts» from implementation and assertions are presented. The approach to verification of event-autonomous nondeterministic units is described. It introduces the dynamic refinement of reference model behavior based on responses from the verified RTL-model. The experience of the application of the methods for Elbrus 8-core microprocessor memory subsystem verification is considered.

Ключевые слова: многоядерный микропроцессор, подсистема памяти, автономная верификация, тестовая система, тестирование на основе моделей, событийная модель, потактово-точная модель, когерентность памяти, событийная недетерминированность.

Keywords: multi-core microprocessor, memory subsystem, standalone hardware verification, test system, model-based verification,

functional model, cycle-accurate model, memory coherence, the event nondeterminism.

Введение

Подсистема памяти современных многоядерных микропроцессоров обеспечивает общий доступ процессорных ядер к данным и согласованность копий данных, которая достигается реализацией протокола когерентности памяти. Подсистема имеет сложную иерархическую структуру, включающую в себя кэш-памяти нескольких уровней (каждая из которых на данном уровне может относиться к одному ядру или быть общей для нескольких ядер), устройства управления доступом к оперативной памяти и пространству ввода-вывода, а также локальные и глобальные справочники устройств подсистемы.

Исходя из официальных перечней (errata), публикуемых производителями микропроцессоров [1, 2], большая доля обнаруженных после выпуска микросхемы ошибок связана с неверным функционированием подсистемы памяти и, в частности, системы поддержки протокола когерентности, причем для фиксации большинства таких ошибок следует одновременно выполнить ряд нетривиальных условий, таких как одновременная нагрузка от всех абонентов, блокировки, специфические пакеты. В связи с этим значительные усилия прикладываются для обнаружения и локализации ошибок этой категории еще до производства микросхемы, в процессе проектирования.

Применительно к многоядерным микропроцессорам наиболее распространенным подходом к верификации подсистемы памяти является ее моделирование в составе целого микропроцессора – системная верификация [3]. В этом случае модель микропроцессора исполняет тестовую программу на языке ассемблера, результаты которой сравниваются с результатами ее работы на эталонной реализации системы команд – архитектурном симуляторе. Проблема данного подхода заключается в том, что компоненты подсистемы памяти зачастую прозрачны относительно исполняемой программы, поэтому трудно провести тестирование, в котором создаются необходимые граничные условия их функционирования,

т.е. обеспечить достаточное качество верификации всей подсистемы. В связи с этим наряду с системной верификацией при проектировании многоядерных микропроцессоров предлагается применять также автономную верификацию наиболее критичных устройств.

В статье рассмотрены основные методы и особенности автономной верификации устройств подсистемы памяти микропроцессоров, предложены подходы к сокращению сроков верификации без потери качества проверки. Описан опыт их применения применительно к подсистеме памяти многоядерных микропроцессоров с архитектурой «Эльбрус».

1. Автономная верификация устройств подсистемы памяти

Объектами автономной верификации устройств подсистемы памяти являются ее компоненты, представленные на одном из языков описания аппаратуры класса HDL на уровне регистровых передач (RTL). Их также называют RTL- или HDL-моделями устройств. Несмотря на разнообразие устройств, входящих в состав подсистем памяти современных многоядерных микропроцессоров, все они обладают рядом общих свойств:

- единицей данных, с которой работает устройство, является кэш-строка;
- запросы, относящиеся к различным кэш-строкам, обрабатываются независимо друг от друга;
- возможна одновременная работа с различными запросами от нескольких абонентов: процессорных ядер, памяти;
- маркировка транзакций (тегированность запросов и ответов);
- каждое устройство подсистемы памяти реализует часть общего механизма поддержки протокола когерентности.

Реализация автономной верификации предполагает генерацию тестовых воздействий, проверку корректности поведения устройства и оценку качества тестирования. Эти задачи решает тестовая система – программа, моделирующая входные воздействия на устройство, анализирующая корректность реакций и оценивающая необходимость даль-

нейшей верификации. Под корректным поведением устройства понимается поведение, соответствующее его спецификации, которая может описывать устройство на различных уровнях абстракции.

2. Генерация воздействий

Генерация воздействий на устройство может быть выполнена в тестовой системе как на уровне интерфейсных сигналов, так и на более высоком уровне абстракции. В номенклатуру интерфейсных сигналов компонента подсистемы памяти может входить до нескольких сотен типов. Соответственно, генерация тестовых воздействий на этом уровне достаточно трудоемка, и при разработке генераторов для автономной верификации сложных RTL-моделей применяют моделирование на уровне транзакций (TLM). Исходя из концепции TLM 2.0, определенной в стандарте [4], TLM – это подход к моделированию реагирующих систем, в котором описание коммуникаций между компонентами отделено от их функциональности. Механизмы коммуникации моделируются каналами, инкапсулирующими низкоуровневые детали передачи данных.

Множество транзакций устройств подсистем памяти можно условно разделить на первичные запросы, реакции от устройства и вторичные запросы, пересылаемые в сообщениях определенного типа. Первичный запрос содержит инструкцию доступа к памяти, реакции – это ответы верифицируемого устройства на первичные запросы, вторичные запросы являются ответами на реакции устройства. Генератор воздействий имитирует окружение устройства, создавая последовательности первичных и вторичных запросов. С каждым первичным запросом req связано подмножество контекстных переменных $\{v_i\}_{req}$ и предусловие – охранный предикат транзакции g_{req}. Значения контекстных переменных и реакций устройства определяют, выполняется ли охранный предикат первичного запроса. Предусловием для вторичных запросов являются только реакции от верифицируемого устройства. Примерами первичных запросов могут служить запросы на чтение из памяти

и запись в память (load/store), вторичных – ответы на снуп-запросы от устройства. В качестве охранных предикатов могут выступать значения сигналов занятости входных буферов устройства, готовности к приёму первичных запросов, готовности устройства (окончания инициализации), а также типичные только для устройств подсистем памяти значения функций, обеспечивающих упорядочивание обращений к памяти и завершённость предыдущих первичных запросов.

Для верификации устройств подсистем памяти генератор воздействий предлагается разделять на две независимые части: генератор первичных запросов, создающий псевдослучайные или направленные последовательности первичных запросов и вычисляющий значения охранных предикатов для каждого запроса, и генератор вторичных запросов, отвечающий за запросы-реакции от устройства. Это даёт возможность более удобной и быстрой отладки общего генератора и построения легко конфигурируемых тестовых сценариев.

3. Проверка корректности

Устройства подсистемы памяти можно отнести к классу параллельных систем обработки и передачи информации. Основные требования, предъявляемые к функционированию параллельных систем, можно свести к следующим свойствам [5]:

- достижимость (reachability) определенных состояний системы;
- безопасность (safety) – невозможность наступления определенных событий;
- справедливость (fairness) – установка, что определенное событие будет выполняться в системе неопределенно часто;
- живость (liveness) – прогресс в исполнении вычислений.

В табл. 1 выделены и соотнесены с соответствующими свойствами основные требования, предъявляемые к корректности функционирования устройств подсистем памяти.

Таблица 1. Классификация требований к функционированию устройств подсистем памяти

№ требования	Требование к функционированию устройства подсистемы памяти	Свойства параллельных систем
1	Запрет приема первичных запросов в случае заполненных буферов приема	Безопасность
2	Соблюдение очередности выполнения первичных и вторичных запросов	Безопасность
3	Выдача прерываний или сообщений об отклонении при некорректных запросах	Достижимость
4	Полная обработка корректных первичных и вторичных запросов	Достижимость / Справедливость / Живость
5	Отсутствие ложных реакций	Живость / Безопасность
6	Предоставление ресурсов всем абонентам в равной или заранее определенной мере	Справедливость
7	Отсутствие взаимных блокировок (deadlock)	Живость / Безопасность
8	Отсутствие «активных тупиков» (livelock)	Живость
9	Выдача реакций соответствующему абоненту	Достижимость

В соответствии с этими требованиями корректность поведения RTL-модели устройства подсистемы памяти можно установить путем двух проверок: *проверки корректности потока* (flow control) и *проверки данных* (data control). К первой из них относятся проверки выполнения требований 1, 4, 7-9 (табл. 1). Под проверкой данных подразумевается проверка содержимого транзакции реакции: определение корректности данных, тегов, идентификаторов запросчиков (требования 2, 3, 5). Далее предлагаются подходы к выполнению проверки перечисленных свойств.

3.1. Сравнение с эталонной моделью

Эталонная модель реализуется по спецификации устройства на языке высокого уровня (языке общего назначения, например, C, C++ или специализированном языке верификации аппаратуры, таком как SystemVeriolog или «e»). Устройство и его эталонная модель одновременно включены в тестовую систему, и расхождение в их реакциях сигнализирует об ошибке.

Если на поведение устройства наложены строгие временны**е** ограничения, то его поведение должно быть специфицировано на уровне регистровых передач, чтобы была возможна разработка его потактово-точной модели. В модель передается информация о переданных и принятых устройством сообщениях, а также о сигнале синхроимпульса. На каждом такте проверяется соответствие сообщений от верифицируемого и эталонного устройств, а несовпадение какого-либо сообщения или его отсутствие служит признаком ошибки.

Достоинствами этого подхода являются:

- быстрое обнаружение ошибок: на следующем такте после расхождения в поведении эталонного и реального интерфейсов;
- значительный объем информации для исправления ошибок в устройстве, эталонной модели или спецификации;
- возможность полной проверки временны**х** свойств устройства.

К недостаткам можно отнести:

- неустойчивость модели к изменениям в устройстве;
- сходство реализаций эталонной модели и устройства (может привести к дублированию ошибок);
- значительная трудоемкость реализация потактово-точной модели.

Если для устройств не установлены жесткие временны**е** ограничения на исполнение инструкций и им свойственно однозначное отображение множества воздействий на множество реакций, то оправдано использование событийных моделей, работающих не на уровне регистровых передач, а на уровне транзакций [6]. При этом подходе необходимо разработать механизмы преобразования интерфейсных сигналов в атомарные транзакции – сериализаторы воздействий и десериализаторы реакций. Такой подход позволяет относительно быстро обнаруживать ошибки при приеме некорректного пакета, дает большую устойчивость модели к изменениям в реализации, однако не позволяет проверять времен-

н***ы***е свойства устройства и не учитывает внутреннюю динамику его работы.

Если на некоторые из функций устройства накладываются временн***ы***е ограничения, можно воспользоваться комбинацией двух подходов. При этом эталонная модель условно разделяется на компоненты, потактово-точно моделирующие критические к времени функции, и компоненты, передающие сообщения на уровне транзакций. Такой гибридный подход во многом лишен недостатков потактово-точных и событийных моделей.

3.2. Применение функциональных утверждений (assertions)

При функциональной верификации для обеспечения корректности внешних воздействий необходимо с некоторой точностью моделировать поведение внешней системы. В этом случае одним из возможных подходов к верификации является анализ состояния компонентов тестовой системы, имитирующих окружение верифицируемого устройства. При этом проверяется выполнение требований к воздействиям на модули окружения, поступающим от тестируемого устройства. Требования протоколов взаимодействия между устройством и его окружением добавляются в качестве функциональных утверждений (assertions) в соответствующие компоненты генератора воздействий. Нарушение данных утверждений является признаком ошибки.

Для устройств подсистемы памяти специфично требование согласованности состояний различных устройств. Для этого при изменении состояния каждого из компонентов тестовой системы происходит анализ согласованности всей системы. Тем самым проверяется как корректность потока, так и корректность данных путем анализа пакетов запросов и ответов, полученных от верифицируемого устройства.

Достоинствами такого подхода являются относительная малая необходимость изменений в тестовой системе (они требуются только при достаточно редких изменениях протоколов взаимодействия между устройствами), а также возможность обнаружения ошибок и несоответствий в документации. Недостатком является ограниченность множества свойств устройства, которые можно проверить, используя только утверждения. Это значи-

тельно снижает возможности использования утверждений в качестве единственного механизма проверки в тестовой системе.

3.3. Подходы к проверке событийно-недетерминированных устройств

Необходимым условием того, чтобы разработка эталонной событийной модели стала возможной, является детерминированность спецификации уровня транзакций. Иначе говоря, существует единственный вариант корректного поведения устройства на уровне транзакций. Однако следует понимать, что на уровне регистровых передач в *событийно-недетерминированных* устройствах идентичные тестовые воздействия приведут к одинаковым тестовым реакциям. Проблема заключается в том, что тестовые воздействия с различными временн***ы***ми характеристиками могут отображаться в единственную трассу событий уровня транзакций, хотя зачастую незначительные временн***ы***е изменения во входных воздействиях могут привести к различным вариантам поведения устройства. В устройствах подсистемы памяти примером подобного поведения может являться алгоритм вытеснения данных из кэш-памяти (как правило, NRU – Not Recently Used) или выбор запроса на исполнение среди нескольких запросчиков.

Существуют несколько вариантов решения этой проблемы. Приведем два из них:

1. Метод «серого ящика». Этот способ заключается в использовании дополнительной информации, полученной от RTL-модели устройства, для определения реального поведения в случае, когда при идентичных воздействиях на уровне транзакций корректными являются несколько вариантов поведения устройства. Для реализации этого подхода необходима доработка спецификации устройства: определение внутреннего интерфейса устройства, где исполняется недетерминированный с точки зрения спецификации уровня транзакций выбор. Информация о сигналах этого интерфейса должна быть передана в качестве «подсказки» в тестовую систему. Такой подход также называют методом «серого ящика».

Анализ «подсказки», поступившей от исследуемой модели, в эталонной событийной

модели помогает определить единственный вариант перехода в устройстве. Применение такого решения обеспечивает относительную простоту реализации и возможность использования «подсказок» в механизмах генерации тестовых воздействий и анализе качества тестирования. При этом требуется расширение спецификации и стабильность состава внутренних интерфейсов устройства, что относится к недостаткам метода.

2. Проверка множеств допустимых состояний устройства. Альтернативным подходом является проверка всего множества возможных вариантов поведения устройства, разрешенных спецификацией. В случае возникновения ситуации недетерминированного выбора, для каждого из вариантов поведения необходимо создать отдельный экземпляр эталонной модели. Если дальнейшее поведение устройства не соответствует поведению экземпляра модели, он завершает свою работу. Признаком обнаружения ошибки в данном случае является завершение работы каждой из эталонных моделей [7].

В общем случае количество вариантов поведения (и, соответственно, количество экземпляров эталонных моделей) может расти экспоненциально росту количества транзакций. Однако для устройств подсистемы памяти возможна эффективная реализация предложенного подхода, поскольку операции в различные кэш-строки независимы, и операции, попадающие в одну кэш-строку, сериализуются. Такой метод устойчив к изменениям реализации устройства и применим в случае, когда внутренние интерфейсы устройства не специфицированы.

4. Опыт практического применения

Описанные методы были применены при автономной верификации устройств подсистемы памяти восьмиядерного микропроцессора с архитектурой «Эльбрус»: глобального справочника (DC, Directory cache), кэш-памяти третьего уровня (L3-cache/L3-кэш), модуля доступа к оперативной памяти (MAU, Memory Access Unit), хост-контроллера (HC).

Для полной функциональной верификации глобального справочника подсистемы памяти была разработана потактово-точная модель устройства. Глобальный справочник

работает синхронно с конвейером в системном коммутаторе процессора, соответственно, требовалась тщательная проверка временны*х* характеристик устройства. Кроме того, устройство включает механизм вытеснения строк по протоколу NRU, для моделирования которого необходима точная информация о времени занесения кэш-строки в справочник.

Метод построения гибридной модели был применен при автономной верификации MAU. Это устройство реализует механизм подклеивания – объединения нескольких схожих запросов чтения из кэш-памяти второго уровня в один при соблюдении ряда условий. В эталонной модели с потактовой точностью воспроизведена функциональность передачи первичных запросов между L2-cache и L3-cache, что позволило проверить реализацию механизма подклеивания. Для прочих функций устройства использовалась функциональная модель, что позволило значительно упростить моделирование компонентов, временны*е* характеристики которых не столь важны.

Для остальных устройств подсистемы памяти использовались событийные модели уровня транзакций, которые значительно проще в разработке, отладке и позволяют проводить автономную верификацию при условиях итеративной разработки RTL-модели и неполноты спецификации. Для достижения приемлемого качества проверки применялись предложенные в разделе 3 дополнительные механизмы и подходы к проверке. При верификации хост-контроллера был успешно использован метод функциональных утверждений, позволивший проверить корректность потока процессорных и DMA обменов между устройством, южным мостом и ядрами процессора.

L3-кэш верифицируемого микропроцессора выполняет запросы от восьми микропроцессорных ядер и системного коммутатора. При этом в спецификации не определено, какой из запросов от различных ядер будет выбран первым для обработки: это поведение определяется внутренней динамикой [8], поэтому L3-кэш является событийно-недетерменированным устройством. Кроме того, он достаточно сложен, поэтому разработка потактово-точной или гибридной моделей очень трудоемка. В связи с этим при авто-

номной верификации в модуле проверки L3-кэша был реализован предложенный в разделе 3.3 метод (Б), а также дополнительные функциональные утверждения, позволившие проверить реализацию протокола поддержки когерентности.

Заключение

Подсистема памяти является важным компонентом микропроцессора, для обеспечения корректной работы которой необходима автономная верификация входящих в нее устройств. С этой целью в рамках тестовой системы устройства реализуют механизмы генерации тестовых воздействий и проверки корректности реакций устройства, зависящие от его специфики. В работе определены задачи генератора воздействий и модуля проверки тестовой системы и описаны подходы к их организации в зависимости от особенностей реализации устройства и полноты спецификации. Кроме традиционных методов применения эталонных моделей разного уровня абстракции предложены методы проверки для событийно-недетерменированных устройств, которыми зачастую являются устройства подсистемы памяти. Комбинированное применение рассмотренных подходов (функциональных утверждений, «подсказок» от исследуемой модели или реализация всех допустимых состояний в эталонной модели) позволяют существенно сократить сроки на разработку тестовой системы и эталонной модели, поддерживая при этом качество проверки, сопоставимое с применением потактово-точных моделей.

Описанные подходы были применены при автономной верификации нескольких устройств подсистемы памяти многоядерного микропроцессора с архитектурой «Эльбрус». В результате был найден и исправлен ряд ошибок в реализациях устройств и эталонных моделях. Эти результаты могут найти применение и при верификации устройств подсистем памяти многоядерных микропроцессоров других архитектур.

Литература

1. Revision Guide for AMD Family 15h Models 30h-3Fh Processors. [Электронный ре-

cypc]. URL: www.developer.amd.com/resources/documentation-articles/developer-guides-manuals (дата обращения 20.12.2015).

2. Intel® Atom™ Processor C2000 Product Family. Specification Update. December 2015. [Электронный ресурс]. URL: www.intel.com/content/www/us/en/processors/atom/atop-c2000-family-spec-update.html (дата обращения 20.12.2015).

3. Стотланд И.А., Куцевол В.Н., Мешков А.Н. Проблемы функциональной верификации кэш-памяти второго уровня микропроцессоров с архитектурой «Эльбрус» // Вопросы радиоэлектроники. – 2015. – Сер. ЭВТ. – Вып. 1. – С. 76-84.

4. TLM-2.0.1. TLM Transaction-Level Modeling Library. [Электронный ресурс]. URL: http://www.accellera.org/downloads/standards/systemc (дата обращения 20.12.2015).

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

6. Стотланд И.А., Лагутин А.А. Применение эталонных событийных моделей для автономной верификации модулей микропроцессоров // Вопросы радиоэлектроники. – 2014. – Сер. ЭВТ. – Вып. 3. – С. 17-27.

7. Камкин А.С., Петроченков М.В. Метод построения тестового оракула для подсистемы памяти микропроцессора на основе недетерминированной функциональной модели // Вопросы радиоэлектроники». – 2015. – Сер. ЭВТ. – Вып.1. – С. 84-95.

8. Кожин А.С., Кожин Е.С., Костенко В.О. Кэш третьего уровня и поддержка когерентности микропроцессора «Эльбрус-4С+» // Вопросы радиоэлектроники. – 2013. – Сер. ЭВТ. – Вып.3. – С. 26-38.