Использование метода косимуляции при разработке высокопроизводительных микропроцессоров

А.Г. Ворсин[1], А.В. Шумаков[2], К.А. Петров[3], П.С. Зубковский[4]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, vorsin@cs.niisi.ras.ru, +7(962)403-51-87;

[2]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, shumakov@cs.niisi.ras.ru, +7(916)235-27-67;

[3]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, petrovk@cs.niisi.ras.ru, +7(926)145-88-93;

[4]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, zubkovsky@cs.niisi.ras.ru, +7(495)719-78-49

Аннотация. Моделирование, проводимое на низком уровне абстракции при разработке высокопроизводительных микропроцессоров, создает большую нагрузку на вычислительные машины и занимает много времени. Рассмотрен метод косимуляции, позволяющий использовать часть моделируемого микропроцессора в виде абстракции более высокого уровня, тем самым сокращая нагрузку на вычислительные машины и время моделирования.

Ключевые слова: система на кристалле (СнК), высокопроизводительный микропроцессор, RTL-моделирование, программная эмуляция, косимуляция

1. Введение

В процессе разработки системы на кристалле (СнК) высокопроизводительного микропроцессора во избежание накопления ошибок важно как можно раньше и как можно более обширно начать тестировать разрабатываемую модель. Тестирование можно проводить различными способами [1-3] в зависимости от конкретных задач, условий и этапа разработки.

Так, например, функциональное RTL-моделирование позволяет отслеживать и изучать исполнение отдельных логических операций проекта на низком уровне абстракции. С другой стороны, этот способ значительно уступает в скорости менее подробным типам моделирования, так что подходит либо для тестирования не слишком больших устройств, на коротких временных промежутках, либо когда у разработчика нет строгих ограничений по времени [4].

Существуют методы тестирования, при использовании которых части одного моделируемого высокопроизводительного микропроцессора существуют на различных уровнях абстракции либо описаны разными языками. Это может быть ПЛИС-прототипирование в комбинации с RTL-моделированием, Моделирование на уровне транзакций и RTL-моделирование, программная эмуляция аппаратуры и RTL. Все эти методы обозначаются термином косимуляция [5]. Этот подход в определенных случаях имеет превосходство в скорости моделирования и удобстве для разработчика, так как позволяет комбинировать разные уровни абстракции.

Метод косимуляции был применен в процессе тестирования контроллера интерфейса 10-гигабитного Ethernet. В данном случае контроллер был представлен в виде модели, описанной на языке RTL и программно подключен к модели СнК эмулируемой на более высоком уровне абстракции в программе QEMU.

Решение провести тестирование контроллера этим методом было обусловлено необходимостью оперативной проверки на совместимость с драйвером для операционной системы Linux, запуск которой на RTL-модели СнК занял бы неоправданно много времени.

2. Актуальность метода косимуляции

При проектировании СнК и ее составных частей используются различные уровни абстракции. Каждый из них имеет свои преимущества на разных этапах. К уровням абстракции, применяемым при логическом проектировании можно отнести уровень логического нетлиста, уровень RTL и уровень программной эмуляции аппаратуры.

При выборе некомбинированного (проводящегося на одном уровне абстракции) метода тестирования следует учитывать преимущества и недостатки каждого из них.

RTL-модель описывает цифровое устройство в виде последовательных логических операций. Такое описание легко воспринимается человеком и может быть напрямую преобразовано в

модель уровня логических элементов с помощью САПР синтеза.

Компиляция модели – то есть ее сборка САПР на основе исходного кода – занимает порядка нескольких минут и включает в себя составление иерархической структуры и проверку синтаксиса языка.

Напротив, моделирование работы устройства на этом уровне подразумевает определение состояния всех его сигналов в каждом такте и может потребовать значительного времени. Соотношение машинного (времени работы вычислительной системы, на которой производится моделирование СнК) и системного (времени в симуляции) времен может достигать 10^8.

С другой стороны, разработчик получает возможность отслеживать ошибки, сужая зону поиска вплоть до логического элемента, на котором они возникают в любой момент времени.

Уровень логического нетлиста получается путем добавления к RTL описанию информации о задержках сигнала, проходящего через логические элементы и между ними. При моделировании он позволяет производить отладку ошибок, которые не проявляются в RTL, однако его компиляция и сам процесс моделирования на несколько порядков медленнее.

Прототипирование модели СнК микропроцессора на ПЛИС требует предварительной подготовки после каждого изменения структуры исходного кода, так как САПР должна синтезировать файл-прошивку. Благодаря тому, что ПЛИС обрабатывает процессы параллельно, ее производительность сопоставима с производительностью реального устройства. Однако для возможности прототипирования продвинутых устройств со сложной структурой требуются дорогостоящие тестовые стенды.

Программная эмуляция позволяет воспроизводить функции одних вычислительных систем на других. Эмуляция дает возможность взаимодействовать с эмулируемым устройством программными средствами, но при этом отсутствует возможность подробного исследования его внутренней структуры.

Сравнительные характеристики методов тестирования модели микропроцессора при разработке представлены в таблице 1.

Таблица 1. Сравнительные характеристики методов тестирования модели микропроцессора при разработке

Методы / Параметры	RTL моделирование	ПЛИС прототипирование	Программная эмуляция
Время компиляции	Малое	Высокое	Малое
Машинное время моделирования	Высокое	Малое	Малое
Подробность отладочной информации	Высокая	Средняя	Малая
Стоимость аппаратуры для моделирования	Низкая	Высокая	Низкая

Сочетание RTL-моделирования и программной эмуляции в рамках косимуляции позволяет перенять сильные стороны обоих способов в виде высокой скорости тестирования, которая незначительно падает по сравнению с чистой эмуляцией, и высокой подробности внутренних процессов отлаживаемой части. При этом не требуется дополнительных затрат на покупку дорогостоящего оборудования в виде стендов ПЛИС, для проведения косимуляции достаточно персонального компьютера. Стоит отметить, что для реализации данного способа требуется дополнительное время на предварительную подготовку, а также наличие у разработчика знания языков Verilog и C.

3. Используемый эмулятор

QEMU это программное обеспечение с открытым кодом, написанное на языке C, для эмуляции архитектур одних систем на других. Программа поддерживает режимы “пользовательской” и “системной” эмуляции. В первом случае эмулируется только центральный микропроцессор, а во втором и все периферийные устройства [6]. Моделирование в QEMU гораздо быстрее RTL-моделирования, поэтому при необходимости тестирования отдельного устройства в со-

ставе системы перенос основной ее части в эмулятор позволит ускорить ход симуляции в сотни раз. При этом, если оставить тестируемое устройство на уровне RTL, можно будет производить его отладку, не создавая дополнительную нагрузку на вычислительную машину, на которой проводится моделирование.

QEMU поддерживает процессорные архитектуры ARM, MIPS, x86, и другие, а также имеет возможность добавления собственных. При косимуляции контроллера 10-гигабитного Ethernet эмулировалась СнК, разработанная в ФГУ ФНЦ НИИСИ РАН с MIPS-подобной архитектурой процессора. В файле, описывающем этот проект для QEMU, обозначены периферийные модули с указанием базовых адресов, адресных окон и линий прерывания. Устройство, участвующее в косимуляции на уровне RTL, здесь обозначается как “cosim”.

Само устройство “cosim” для QEMU выглядит как описание типов возможных транзакций. Всего 5 типов:

1. Передача запроса тестируемому устройству, с указанием его типа. Запрос может быть чтением или записью в регистр контроллера, или сбросом. В случае записи, помимо адреса целевого регистра контроллеру передадутся данные. Запрос типа сброс приведет к аппаратному сбросу контроллера.
2. Ответ от контроллера. Приходит после каждой передачи запроса и свидетельствует о том, что запрос обработан.
3. Прямой доступ в память (DMA) для записи, где контроллер выступает в роли мастера.
4. DMA чтение из памяти, где контроллер выступает в роли мастера.
5. Передача сигнала прерывания от контроллера.

Описание транзакций выполнено в виде функций, вызываемых согласно алгоритму драйвера или бареметального теста.

4. Обеспечение связи между ПО и RTL-моделью

Связь QEMU с тестируемым устройством осуществлялась посредством интерфейса DPI (direct programming interface) [7]. Этот интерфейс специально предназначен для взаимодействия языков проектирования Verilog и SystemVerilog с языками программирования. Он был реализован в виде вызова функций, подключаемых с помощью библиотеки libdpi.so, которые распознаются как эмулятором QEMU, так и компилятором RTL-кода.

Схема взаимодействия RTL-модели моделируемого контроллера и эмулируемой модели СнК представлена на Рис. 1. Функции библиотеки libdpi.so помещают получаемые при вызове данные в FIFO файлы, которые еще называют “named pipe” каналами. Они предварительно создаются в директории симуляции командой makefifo. На каждое устройство в косимуляции, создается 4 таких канала: для чтения и записи инициализируемых QEMU, и чтения и записи инициализируемых контроллером. Такое количество каналов необходимо для того, чтобы при чередовании различных типов передач данные не перемешивались.

Со стороны QEMU с библиотекой libdpi.so взаимодействует модуль “extcom”. Его копии подключаются отдельно для каждого косимулимуемого устройства и выполняют преобразование формата данных между библиотекой и QEMU, а также помещают и забирают данные из FIFO.

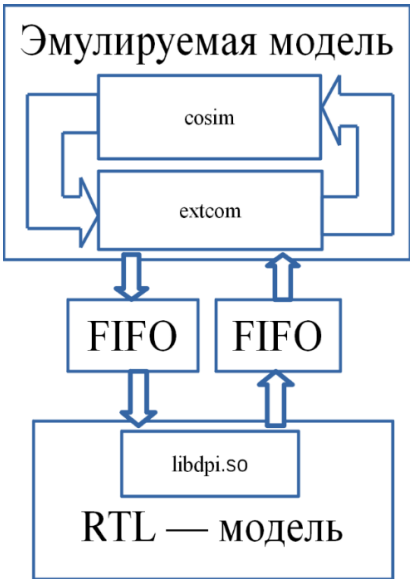

Рис. 1. Схема взаимодействия RTL-модели и эмулируемой модели СнК

5. Подготовка RTL к косимуляции

Для возможности взаимодействия с QEMU, со стороны RTL был создан модуль-обертка в который импортируются функции из libdpi.so. В этот модуль помещается модуль верхнего уровня контроллера, а сигналы его интерфейса взаимодействия с регистрами управляются поведенческими блоками синхронной логики, где выполняется преобразование данных между интерфейсом и функциями библиотеки. Таким образом взаимодействие QEMU и контроллера синхронизируется по тактовой частоте, определяемой на уровне модуля обертки.

6. Результаты

В качестве эксперимента был составлен бареметальный тест, в котором контроллер 10-гигабитного Ethernet работает в режиме петли с прямым доступом в память, результаты которого для двух методов моделирования (RTL-моделирование и косимуляция) представлены в таблице 2.

Таблица 2. Сравнение методов моделирования

	RTL моделирование всей СнК	Косимуляция RTL (контроллер) + QEMU (СнК)
Системное время, нс	367	19
Машинное время моделирования, с	1491	4
Объем задействованной оперативной памяти ЭВМ, МБ	1483	67

Данные из таблицы 2 были получены средствами САПРа. Для данного блока машинное время, необходимое для завершения теста в косимуляции меньше времени при RTL моделировании примерно в 350 раз, при том, что максимальный объем оперативной памяти занятый САПРом для моделирования меньше в 22 раза. Стоит отметить, что память, которая потребовалась для запуска QEMU не учитывалась.

Были успешно завершены тесты, затрагивающие различный функционал контроллера, а также отданы команды на передачу пакетов данных с помощью драйвера для операционной системы Linux.

7. Заключение

В настоящее время возрастающая сложность и размер структур СнК высокопроизводительных микропроцессоров требуют как можно более подробного их тестирования на ранних этапах проектирования. В связи с этим возрастает время тестирования, а также требования к вычислительным машинам, связанные с повышенной нагрузкой на них.

Косимуляция позволяет на ранней стадии разработки СнК начать отлаживать взаимодействие между аппаратным и программным обеспечением. Производительность этого метода позволяет использовать для тестирования сложное ПО (вплоть до запуска операционной системы).

В тоже время, исполнение отлаживаемой части в виде RTL-модели позволяет производить детальную отладку тестируемого устройства, а повторная компиляция при внесении изменений в аппаратный код не занимает много времени.

Было успешно проведено тестирование RTL-модели контроллера интерфейса 10-гигабитного Ethernet в составе СнК, эмулируемой программой QEMU. В процессе тестирования был достигнут 350-кратное снижение машинного времени моделирования, а также 22-кратное снижение потребляемой программой симулятором оперативной памяти.

Публикация выполнена в рамках государственного задания ФГУ ФНЦ НИИСИ РАН по теме FNEF-2022-0004.

Using the Cosimulation Method in High-Performance Microprocessors Development

A.G. Vorsin, A.V. Shumakov, K.A. Petrov, P.S. Zubkovskiy

Abstract. In the development of high-performance microprocessors, low abstraction level modeling creates a large load on computing machines and takes a lot of time. A simulation method that allows to reduce load on computers and simulation time using a part of the simulated microprocessor in the form of a higher-level abstraction is considered in this article.

Keywords: System on Chip (SoC), high-performance microprocessor, RTL-modeling, program emulation, cosimulation.

Литература

1. Robert G. Sargen, Verification and validation of simulation model, IEEE 2011.
2. Harry D. Foster, 2018 FPGA Functional Verification Trends, IEEE 2018.

3. Ney Calazans, Edson Moreno, Fabiano Hessel, Vitor Rosa, Fernando Moraes, Everton Carara, From VHDL Register Transfer Level to SystemC Transaction Level Modeling: a Comparative Case Study, IEEE 2003.

4. Maddu Karunaratne, A. Sagahayroon, RTL fault modeling, IEEE 2005.

5. Claudio Gomes, Casper Thule, David Broman, Peter Gorm Larsen, Co-Simulation: A Survey, 2018 ACM Computing Surveys.

6. Fabrice Bellard, QEMU, a Fast and Portable Dynamic Translator, DBPL 2005.

7. IEEE Standard for SystemVerilog — Unified Hardware Design, Specification, and Verification Language, 901-915, IEEE 2013.