УДК 004.4.244

Гибридное моделирование систем на кристалле с трансляцией кода функционального описания в RTL “на лету”

М.Н.Алекперов[1], А.В.Селютин[1], Ю.Н.Фонин[1], В.В.Ахлынин[1]

[1]Московский физико-технический институт (государственный университет)

В работе рассматривается методология проектирования цифровых электронных схем с применением высокоуровневого моделирования компонентов системы на SystemC, обосновывается эффективность TLM для этапов функционального описания, верификации и событийной (event-driven) симуляции модели. Далее показывается синтаксическая близость SystemC и Verilog и предлагается решение для трансляции соответствующих структур и классов окружения SystemC в модули Verilog. Предсказывается возможность частичного использования объектно-ориентированных возможностей языка C++, предлагается механизм частичной поддержки наследования, полиморфизма, шаблонов, каналов и абстрактных интерфейсов.

С ростом допустимой плотности размещения элементов и неумолимым развитием технологий проектирования систем на кристалле стало возможным разрабатывать цифровые устройства со сложной архитектурой, включающей в себя множество подмодулей. Для подобных схем был необходим новый подход на этапе разработки/верификации: разработка и симуляция системы непосредственно на уровне RTL, несмотря на ее достоверность (pin-accurate, register-accurate, cycle-accurate, precise timed), очень времязатратна. Помимо того, что само моделирование RTL-модели на несколько порядков медленнее функционального моделирования, модификация, отладка и повторное использование подобного кода при таком подходе существенно затруднены.[1]

Решением, сочетающим в себе достоинства высокоуровневого моделирования и временное соответствие (cycle accuracy) модели объявленному функциональному описанию является использование высокоуровневого синтеза (HLS) и, как части процесса разработки, моделирования уровня транзакций (transaction level modeling или сокращенно TLM). [2] Одним из ключевых удобств высокоуровневого моделирования является то, что данная технология позволяет преодолеть пропасть между алгоритмическим и схемотехническим проектированием, стирая границы и открывая перед собой мир проектирования hardware разработчикам, до этого не имевшим дела с HDL. Намного проще и эффективнее при разработке RTL-описания опираться на существующую функциональную модель, нежели переписывать систему “с нуля” на совершенно другом языке. Разработка алгоритма без привязки к конкретной hardware платформе придает процессу гибкость; помимо прочего, на этом уровне абстракции нежелательно погружать программиста в тонкости проектирования под данный target - помимо временных затрат, это приведет к сужению взгляда и невозможности перенести решения на другие платформы. [3]

При использовании высокоуровневого моделирования, в частности, TLM, процессы описания, функционального и системного(первичного) моделирования и функциональной верификации сливаются благодаря использованию “золотой архитектуры” = архитектуры, сочетающей в себе частичную (поцикловую/регистровую) HW достоверность (cycle-accuracy, register-accuracy, approx. timed), возможность моделирования с использованием стандартных (fifo, mutex) интерфейсов и использованием hw-типов. При этом, взаимодействие между процессами на этом уровне для удобства описывается вызовами функций.

Одним из принятых стандартов TLM моделирования является SystemC - библиотека, поддерживающая множество специальных типов, применяемых в описании архитектур, а не в программировании, стандартные (fifo, mutex) интерфейсы. Быстрая симуляция, возможность повторного использования кода, возможность применения объектно-ориентированного подхода сделали SystemC де-факто стандартом TLM.[4][5] SystemC является хорошей базой для реализации инструментов высокоуровневого синтеза: он сочетает в себе преимущества объектно-ориентированного подхода, унаследованного от C++, с поддержкой таких низкоуровневых конструкций, как сигналы, порты и возможностью описать архитектуру системы и ее иерархию.

При тестировании большую часть ошибок можно найти на высоком уровне абстракции, при моделировании системы на уровне поведения (behavioural modeling). Это позволяет сэкономить время проектирования и ускоряет процесс отладки RTL-модели. Низкоуровневое моделирование требует на порядок больше времени, чем поведенческое, и высокоуровневый синтез является возможностью отловить существенную часть возможных проблем на функциональном уровне абстракции.[5]

После проведения разработки на системном уровне, отладки и функциональной верификации, разработка переходит в стадию перевода TLM-описания системы в RTL-модель. Инструменты высокоуровневого синтеза решают bottleneck, связанный с переводом функционального описания модели в схематическое (RTL). При автоматизации этого процесса (а, как следствие, и процесса тестирования) можно обеспечить существенный выигрыш в гибкости и скорости разработки SoC. [6]

Не секрет, что с точки зрения синтаксиса языки C/C++ и Verilog весьма близки. Исходя из этого, можно предсказать возможность трансляции исходного кода в язык описания схем. Это является основой инструментов высокоуровневого синтеза. Весьма подробный обзор существующих решений HLS можно найти, например, в [3].

В предложенной реализации транслятора акцент сделан на частичную поддержку иерархичности и объектно-ориентированного функционала SystemC. Транслятор представляет собой приложение, основанное на возможностях фронтенда clang (CFE - Clang FrontEnd). Выходным файлом транслятора является переведенное на язык Verilog описание SystemC-модели.

Основой для построения модели внутреннего представления SystemC-иерархии взят уже известный сообществу проект systemc-clang. Он представляет собой набор классов, выполняющих обход специфических узлов синтаксического дерева программы, и хранящих информацию о модели SystemC и взаимоотношениях/содержимом модулей.

Трансляция модели выполняется сверху вниз. Начиная с верхнего модуля, в функции sc_main просматриваются инициализации других модулей SystemC. На базе sc_main создается модуль тестирования, где описывается структурная схема верхнего уровня, генерируется тестовый синхросигнал и может быть реализовано и само тестирование. Выходной файл создается путем кодогенерации с использованием инфраструктуры представления модели. Модель может состоять из нескольких модулей, каждый экземпляр которого транслируется в RTL-описание. Корректность полученного кода проверяется при помощи IcarusVerilog и Xilinx Vivado. Для симуляции разработчками использовались Vivado и IcarusVerilog+vvp.

Связка инфраструктуры LLVM и Clang является мощным инструментом для построения инструментов анализа кода C++. Именно это позволяет эффективно задействовать возможности объектно-ориентированной парадигмы программирования при трансляции на язык уровня RTL.[3] С целью воплотить данную идею в жизнь в генерацию промежуточного представления systemc-clang и инфраструктуру верхнего обходчика были внесены существенные дополнения и изменения. Так же, инструмент дополнен модулем кодогенерации для языка Verilog.

Краеугольным камнем предлагаемого продукта является демонстрация таких возможностей C++ (и, как следствие, SystemC), как наследование, возможность объявления абстрактных методов (поддержка виртуальных функций), возможности метапрограммирования (шаблонизация и параметризация, макросы), а также поддержка sc_interface - абстрактных каналов с настраиваемыми пользователем методами обработки данных. Помимо дополнения инструмента systemc-clang объектно-ориентированными нововведениями, в трансляторе де-факто была налажена возможность доступа к методам класса (и модели, в том числе), реализована поддержка стандартных C-типов (int, double, etc..) в модулях (раньше были поддержаны лишь стандртные - sc_int, sc_uint - переменные).

Литература

1. HLS Blue Book,Mentor Graphics Corporation, All Rights Reserved
2. An Introduction to High-Level Synthesis,Philippe Coussy Universite´ de Bretagne-Sud, Lab-STICC Michael Meredith Forte Design Systems
3. An overview of today’s high-level synthesis tools. Wim Meeus. Kristof Van Beeck. Toon Goedemé. Jan Meel. Dirk Stroobandt
4. An Efficient TLM/T Modeling And Simulation Environment Based On Conservative Parallel Discrete Event Principles Emmanuel Viaud, Francois P ̂echeux, Alain Greiner
5. An Introduction to High-Level Synthesis, P.Coussy, M.Meredith, 2009, IEEE
6. An Open-Source Tool For Systemc To Verilog Automatic Translation. J. Castillo, P. Huerta And J. I. Martínez.