Обратное проектирование FPGA

Черемисинов Д.И.
Объединенный институт проблем информатики НАН Б
Минск, Беларусь
e-mail: cher@newman.bas-net.by

***Аннотация* — Процесс обратного проектирования (reverse engineering) интегральных схем считается инверсией обычного процесса проектирования, то есть по готовому изделию строится его функциональное представление высокого уровня, призванное облегчить понимание работы устройства. Обсуждается применение обратного проектирования при переносе проекта из технологии FPGA в ASIC.**

Ключевые слова: обратное проектирование; FPGA; XDL; ASIC

I. Введение

Процесс обратного проектирования является существенной частью создания конкурентоспособной продукции, и обычно служит средством разработки устройств, более эффективной, чем имеющиеся у конкурентов. Побочная область применения обратного проектирования – перепроектирование на современной базе устарелых компонент, находящихся в составе долговечном оборудовании, такого как военные или космические системы, ядерные реакторы, авиалайнеры, и морские суда. В полупроводниковой промышленности, заказчиками перепроектирования являются или те, кто интересуется технической информацией об устройстве, или те, кто интересуется компонентами устройства, защищенными патентом. Технические информация извлекается обычно внутри фирм-изготовителей прототипа, с целями разработки нового изделия, или стратегического маркетинга или оценки проектных решений конкурентов.

Главные особенности такого проектирования можно получить, сравнивая обратное проектирование с обычным. Обычная разработка – это процесс превращения спецификации в продукт, удовлетворяющий этой спецификации. Между спецификацией и продуктом находятся процессы разработки и изготовления, в которых необходим некоторый человеческий творческий потенциал и автоматизация. Процесс разработки удобно определять в терминах «уровней» абстракции. Считается, что разработка включает построение следующих представлений продукта:

- Письменная функциональная спецификация для человеческого потребления – описание «верхнего уровня»;
- Абстрактное структурное описание (например, исходный текст или диаграмма схемы), для человеческого потребления (обычно машиночитаемое) – описанием «промежуточного уровня»;
- Детализированное структурное описание (например ассемблер, или сеть элементов) для машинного потребления, но возможно постижимое людьми – описание «низкого уровня»; и
- Продукт (FPGA, программируемая кодом настройки, интегральная схема или монтажная плата), строение которых обычно не воспринимается невооруженными органами чувств человека – самый низкий уровень.

При использовании САПР описания промежуточного уровня в форме, воспринимаемом человеком, не нужны, и строятся только в документационных целях. Форма исходной спецификации определяет структуру продукта и может быть выбрана несколькими различными способами, в зависимости от используемой целевой платформы, типа САПР и типа полупроводниковой технологии (например. CMOS или NMOS).

II. Обратное проектирование

Обратное проектирование в общем случае состоит из следующих стадий:

- анализ продукта;
- извлечение описания продукта промежуточного уровня;
- анализ описания продукта интеллектом человека, для построения новой спецификации;
- разработка нового продукта, используя построенную спецификацию.

Обратное проектирование является инверсией обычной разработки в смысле порядка процесса преобразований; его задача заключается в построении спецификации, анализируя продукт. Результат обратного проектирования не гарантирован, в общем случае не возможно даже в теории построить оригинальную спецификацию, изучая только продукт. Обратное проектирование является трудным и трудоёмким, но сейчас оно упрощается благодаря информационным технологиям по двум причинам:

- Во-первых, поскольку методики самого проектирования становятся более формализованными, все большая часть работы по проектированию происходит внутри компьютера. Таким образом, возникает возможность распознавания блоков программы, или группы элементов схемы на основании знания преобразований, производимых компьютерной программой САПР, одной и той же в разных компаниях. Внутреннюю структуру в этом случае легче исследовать и интерпретировать, чем в случае, когда продукт построен полностью интеллектом человека.
- Во-вторых, методики искусственного интеллекта для распознавания образов в исследовании и интерпретации, достигли уровня, где это распознавание структуры в продукте можно выполнить автоматически.

Однако, несмотря на то, что возможно автоматизировать извлечение высокоуровневого структурного описания, осмысление этого описания требует человеческого интеллекта, так как оно строиться с целью дать человеку возможность понять, как функционирует устройство. Это понимание необходимо для разработки новой спецификации, обеспечивающей эффективную реализацию в другом технологическом базисе.

Разработка новой спецификации при перепроектировании FPGA поддается автоматизации, если возможно использование промежуточных описаний, созданных при прямой разработке продукта.

III. Юридический аспект

Часто возникающий вопрос об обратном проектировании – «действительно ли при перепректировании не происходит нарушения законов?» В области разработки полупроводниковых приборов об обратном проектировании в законе США «Semiconductor Chip Protection Act» говорится, что допускается «обратное проектирование масок или схем с целью обучения, анализа, или оценки решений или методик…». Подобное законодательство имеет Япония, Европейский союз, и другие страны [1].

IV. Формат XDL

Полное внутреннее состояние конфигурации памяти программируемых логических устройств FPGA содержится в программном файле, называемом bitstream. Это состояние при функционировании FPGA обычно хранится в памяти типа SRAM и загружается из энергонезависимого устройства памяти при каждом включении питания FPGA. В мире программного обеспечения изготовителями процессоров общепринято публиковать формат потока двоичных сигналов их продукта в руководстве архитектуры для использования при разработке компиляторов. В мире программирования FPGA, ситуация противоположна. В настоящее время ни один изготовитель FPGA не опубликовал формат bitstream своих устройств. Хуже того, чтобы защитить авторские права разработчиков при передаче устройства заказчику, файл программирования FPGA шифруется. Такой кодированный bitstream и загружается из энергонезависимого устройства памяти при включении питания FPGA.

Задача декомпиляции bitstream, извлеченного из SRAM в готовом устройстве, безнадежна. Но если перепроектирование выполняется организацией разработчиком оригинального устройства на FPGA (например, с целью использования более дешевой в массовом производстве технологии изготовления), то задача декомпиляции может быть решена. Задача декомпиляции bitstream возникает и при оригинальном проектировании, например, САПР Xilinx ISE имеет методику проектирования с использование hardware macro (предварительно размещенные блоки в матрице элементов FPGA вместе с внутренними межсоединениями). САПР Xilinx ISE обеспечивает проектировщику возможность преобразования блока, скомпилированного до уровня bitstream, в сеть элементов FPGA. Таким образом, декомпиляция FPGA bitstream становится возможной, но только с использованием инструментов, составляющими собственность изготовителя FPGA.

САПР Xilinx ISE обеспечивает декомпиляция FPGA с помощью формата XDL. Формат XDL (Xilinx Design Language) – это текстовое представление структуры запрграммированной FPGA: компиляция и декомпиляция осуществляется программой *xdl*, имеющейся в последних версиях Xilinx ISE.

Программа *xdl* имеет опции для того, позволяющие построить файлы отчета (с расширением .XDLRC), которые содержат описательную информацию о запрограммированной FPGA Xilinx. Файлы отчета имеют формат, отличный от XDL (поскольку они описывают FPGA, а не проектируемое устройство) и содержат огромный объем информации (несколько гигабайтов текста), описывающие всю сетку элементов запрограммированной Xilinx FPGA.

Формат XDL официально не опубликован Xilinx. Краткое описание с элементарной информации о XDL содержится в «Руководстве пользователя PlanAhead» [2]. Однако формат текстовый и достаточно прост, чтобы описание можно было понять. Кроме того, описание содержит комментарии, которые помогают пользователю в интерпретации файла. Информация, содержащаяся в файле Xdl, полностью описывает настройку конкретной FPGA для выполнения функций проектируемого устройства [3]. Для понимания функционирования устройства по описанию в XDL формате необходимы функциональные описания элементов и коммутационных блоков конкретного типа FPGA. Структурные описания этих элементов и непрограммируемые соединения можно найти в файле отчета.

Файл XDL содержать два типа конструкций, блоки (instances) и цепи (nets). Блок может быть любым логическим элементом в FPGA, например логическим элементом сетки FPGA (slice), блоком ОЗУ, или процессором цифровой обработки сигналов (DSP block). Описание цепи содержит название цепи и компонентов блоков, которые связаны этой цепью.

Каждый элемент сетки FPGA (slice) имеет описание в Xdl формате. Настройки компонентов внутри описаний элементов везде имеют один и тот же вид: name::#parameter. Названия (name) компонентов настройки приведены в описании FPGA. Значение параметра (parameter), отличное от #OFF указывает, что соответствующий компонент используется.

Для извлечения структуры устройства, пригодной для перепроектирования FPGA в другом технологическом базисе, требуется разработка анализаторов формата XDL и файла отчета. Результат работы анализатора XDL должен быть представлен в виде структурного описания, которое может служить исходными данными для нового проектирования.

[1] Musker D. C. Reverse Engineering / D. C. Musker // Protecting & Exploiting Intellectual Property in Electronics, IBC Conferences, 10 June 1998. – [Электронный ресурс]. – 1998. – Режим доступа :http://www.jenkins-ip.com/serv/serv_6.htm – Дата доступа : 01.05.2006.

[2] PlanAhead User Guide.– Xilinx Inc., 2100 Logic Drive, San Jose CA 95124, v8.2 edition, August 21st 2006. – 560 с.

[3] H¨ubner T. Real-time LUT based network topologies for dynamic and partial FPGA selfreconfiguration / M. H¨ubner, T. Becker, J. Becker // Proceedings of the 17th symposium on Integrated circuits and system design (SBCCI 04). – 2004. – P. 28–32.