Декомпиляция программирования элементов LUT в FPGA

Черемисинов Д.И.
Лаборатория логического проектирования
Объединенный институт проблем информатики Национальной академии наук Беларуси
Минск, Республика Беларусь
e-mail: cher@newman.bas-net.by

***Аннотация*—Рассматривается один из этапов задачи проектирования заказных СБИС, когда в процесс проектирования включена разработка FPGA. Этап заключается в построении функционального описания устройства по структурному представлению FPGA Spartan-3 в формате XDL. Показано, что расшифровка функции элемента LUT в FPGA, работающего в режиме сдвигового регистра, требует дополнительной обработки структурного описания после распаковки макроячеек.**

Ключевые слова: проектирование СБИС; декомпиляция конфигурационного файла FPGA

I. Введение

По сравнению с заказными СБИС (ASIC) проектирование для программируемых пользователем вентильных матриц (FPGA) дешевле (нет невозвращаемых затрат – non-recurring engineering charges, NRE), быстрее и с меньшими рисками разработки. В то же время FPGA экономически не выгодны для массового производства дешевых устройств. Поэтому FPGA часто используются как средства прототипирования ASIC-проектов [1]. Создание FPGA прототипа позволяет снизить NRE ценой усложнения маршрута разработки ASIC включением маршрута проектирования FPGA и этапа перенацеливания процесса проектирования FPGA на ASIC (или наоборот ASIC на FPGA).

Главное отличие между FPGA и ASIC – архитектура. FPGA состоит из массива логических блоков (tile), различающиеся функциональным назначением (конфигурируемые логические блоки, память и блоки умножения), и из программируемой структуры коммутации (routing fabric), которая позволяет программировать связи между блокам. Конфигурируемый логический блок (CLB) в свою очередь состоит из нескольких логических макроячеек (slice), в частности CLB FPGA Spartan-3 содержит 4 логических макроячейки. Макроячейку Spartan-3 составляют два функциональных генератора LUT (Look Up Table) с 4 вводами, двухбитовый полный сумматор и два триггера. ASIC имеют более ограничительную структуру, состоящую из нескольких ПЛМ или более мелких комбинационных блоков, связанных с относительно небольшим числом синхронных регистров.

Случай, когда в маршруте проектирования первой разрабатывается FPGA, потенциально более выгоден для снижения NRE. Тогда перенацеливание FPGA на ASIC представляет собой *перепроектирование* (reverse engineering) FPGA. Перепроектирование является инверсией обычной разработки в смысле порядка процесса преобразований; его задача заключается в построении спецификации, анализируя продукт. В FPGA функция устройства не определена во время изготовления, и прежде чем использоваться в схеме, FPGA должна быть запрограммирована, то есть, задана его конфигурация в виде последовательности конфигурационных битов (bitstream). Обратное проектирование FPGA заключается в декомпиляции bitstream или описания, из которого непосредственно строится bitstream.

II. Распаковка результатов проектирования FPGA

В процессе проектирования устройств на основе Xilinx FPGA результирующее структурное описание представлено в формате NCD (Native Circuit Description) [2], описание которого является технологическим секретом Xilinx. Для доступа к структурному представлению на этом уровне Xilinx предлагает текстовый формат XDL, частично описанный в [3]. Формат XDL является плоским структурным описанием в виде списка цепей, связывающих макроячейки FPGA. Для перепроектирования на ASIC нужно из структурного описания в формате XDL получить функциональную спецификацию. Это преобразование можно выполнить в два этапа: на первом строится структурное описание из более мелких блоков, на втором – функциональное описание, являющееся композицией функций этих простых блоков [4]. Первый этап – это операция распаковки макроячеек. Смысл распаковки состоит в построении описания, которое оставаясь структурным, позволяет получить функциональное описание. В распакованном описании название типа элементов позволяет определить отношение вход-выход элемента. Описание каждой макроячейки в формате XDL содержит строку конфигурации, описывающей ее настройку, и распаковка состоит в анализе ее строки конфигурации.

Распакованное структурное описание превращается в функциональное путем дополнения функциональными описаниями составляющих элементов. Соответствующее преобразование – это оценка (evaluation). Для комбинационных элементов макроячейки логическая функция определяется названием элемента.

III. Декомпиляция программирования LUT

Роль основного логического элемента в FPGA играет логическая таблица (LUT – look-up table), представляющая собой однобитное ОЗУ на 16 ячеек. Эта компонента имеет сложное внутреннее устройство: LUT может работать в режиме логической таблицы, в режиме сдвигового регистра, или в режиме оперативной памяти. Режим работы LUT задается в строке конфигурации значением параметра программирования: «#LUT» , «#SHIFT-REG», или «#RAM». Элемент LUT является комбинационной схемой, если значение параметра его программирования есть «#LUT». В этом случае его функция задана в строке конфигурации.

LUT может быть сконфигурирована как 16-битовое синхронное ОЗУ. Две соседних LUT могут быть сконфигурированы как 16-битовое двухпортовое ОЗУ с записью и чтением по одному адресу и чтением по другому адресу. При этом для реализации синхронного режима записи входной бит данного, сигнал записи и адрес запоминаются в триггерах, а для чтения по второму адресу из блока второй LUT используется только мультиплексор чтения.

Можно конфигурировать LUTы как сдвиговые регистры. Это дает огромную экономию по сравнении со сдвиговыми регистрами, построенными из триггеров: одна LUT может реализовать 16-битовый сдвиговый регистр. Целая макроячейка Spartan-3 может реализовать 32-битовый сдвиговый регистр. Когда LUT используется в режиме SHIFT-REG, его структура представляет собой 16 разрядный сдвиговый регистр, связанный с мультиплексором (Рис. 1).

В распакованных макроячейках работу LUT в режиме сдвигового регистра обеспечивают еще несколько элементов, показанных на рис. 2. На этом же рисунке показано соответствие сигналов макроячейки сигналам сдвигового регистра.

В качестве алгоритмического базиса для решения задачи распаковки формата XDL удобно использовать подстановки графов [4], основой которых являются операции поиска и замены подграфов. Подстановка графов задает правило переписывания графа в форме «образец-действие».

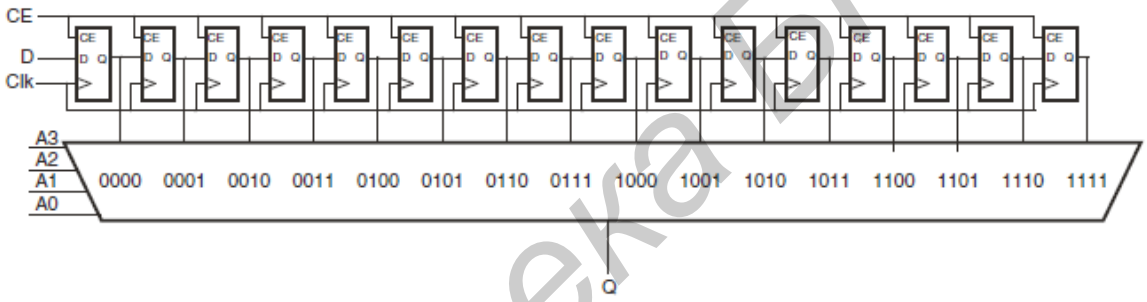

Рис. 1. LUT, реализующая сдвиговой регистр

Элементы макроячейки Spartan-3, обеспечивающие работу в режиме сдвигового регистра Для построения структурной модели, учитывающей программирование LUT в режим сдвигового регистра, требуется структуру FPGA после распаковки преобразовать операцией подстановки графов с образцом, построенным на основе рис.2, и процедурой, состоящей в приклеивании структуры на рис. 1.

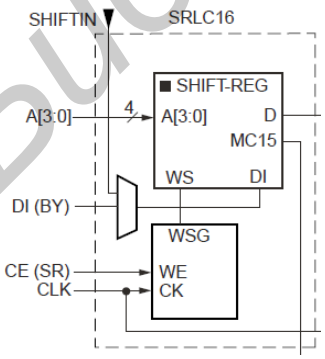

Рис. 2. Элементы макроячейки Spartan-3, обеспечивающие работу в режиме сдвигового регистра

LUT в режиме сдвигового регистра реализует линию задержки на 1-16 тактов синхронизации. Число тактов задержки задается динамически значениями сигналов на линиях A0-A3. LUT может представлять собой и сдвиговой регистр фиксированной длины, если значения сигналов на линиях A0-A3 заданы константами. При построении функционального описания этот случай должен рассматриваться особо, так как использование в качестве функциональной модели структуры на рис. 1 ведет к большой избыточности описания. Однако учет этой особенности ведет к тому, что этап оценки из простого текстового преобразования превращается в сложную систему символических вычислений значений сигналов.

[1] Долинский, М. «Горячие темы» EDA-индустрии по материалам новостей портала DACafe.com // Компоненты и технологии, № 7, 2004. – С. 130-134.

[2] Зотов, Ю.В. Проектирование цифровых устройств на основе ПЛИС фирмы XILINX в САПР WebPack ISE / Ю.В. Зотов. – М. : Горячая линия – Телеком, 2003. – 624 с.

[3] Beckhoff, C. The Xilinx Design Language (XDL): Tutorial and use cases / Beckhoff C., Koch D., Torresen J. // Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2011 6th International Workshop on – 2011. – p. 1-8.

[4] Черемисинов, Д.И. Генерация выполнимых спецификаций из структурных описаний // Инфориатика, № 3, 2012. – pp. 271–350.