DOI 10.17223/20710410/24/10

УДК 621.374.3; 004.4:004.9

О ЗАЩИТЕ ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ В ПРОЦЕССЕ ПРОЕКТИРОВАНИЯ УСТРОЙСТВ НА ОСНОВЕ FPGA XILINX

Д. И. Черемисинов

Объединённый институт проблем информатики НАН Беларуси, г. Минск, Беларусь

E-mail: cher@newman.bas-net.by

Рассматривается влияние задачи охраны интеллектуальной собственности на процесс проектирования устройств на основе FPGA Xilinx. Лицензионное соглашение САПР FPGA Xilinx запрещает использование инструментов этой САПР для перепроектирования, поэтому обсуждаются программно-технические средства обеспечения контроля соблюдения таких соглашений пользователями САПР. Показано, что существует класс устройств на основе FPGA, конфигурационный файл которых принципиально не поддаётся декомпиляции. Этот класс составляют, в частности, устройства с частичной динамической реконфигурацией посредством внутреннего контроллера самореконфигурации.

Ключевые слова: *перепроектирование, FPGA, защита интеллектуальной собственности.*

Введение

Результаты интеллектуальной деятельности имеют стоимостные оценки, как и прочие продукты человеческого труда, так как они могут быть включены в товарооборот на коммерческих условиях и давать полезный эффект (экономический, социальный и т. п.). Защита современных цифровых электронных устройств необходима для предотвращения экономических потерь от кражи интеллектуальной собственности, воплощённой в функциональных возможностях устройства. Возможность кражи обеспечивается при копировании устройства (cloning) или путём его перепроектирования (reverse engineering). В то время как при копировании целью является создание возможно более точной копии оригинального устройства, при перепроектировании нарушитель извлекает алгоритм, реализованный в устройстве, и затем улучшает, изменяет и маскирует его при сохранении функционирования.

Программируемое устройство — это электронный компонент, позволяющий строить реконфигурируемые цифровые схемы. В отличие от элементов с фиксированной функцией, функция реконфигурируемого элемента не определена во время изготовления, и прежде чем использоваться в схеме, элемент должен быть запрограммирован, то есть должна быть задана его конфигурация в виде последовательности конфигурационных битов (`bitstream`).

Программируемые устройства делятся на энергонезависимые и энергозависимые. Энергозависимые устройства не сохраняют битов конфигурации, и конфигурационный файл нужно загружать из внешней памяти при каждом включении питания. В FPGA конфигурационный файл обрабатывается логикой программирования, которая сама не программируема. Реконфигурируемую часть FPGA составляет пользовательская логика (user logic) — она реализует заданное пользователем поведение устройства. Большинство современных FPGA являются энергозависимыми.

Продуктом интеллектуальной деятельности при проектировании FPGA является информация о способах достижения требуемого поведения разработанного устройства (пользовательская логика). Сама информация имеет нематериальный характер, и интеллектуальным продуктом она становится, будучи воплощённой в объективированную форму — техническую документацию, тесты, описания на формальном языке, которые служат исходными данными для САПР и т. д. Опасность — возможность возникновения ситуации, при которой можно понести ущерб. Информационная безопасность обеспечивается защитой информации. Защита информации должна обеспечивать предотвращение ущерба в результате утери (хищения, утраты, искажения, подделки) информации в любом её виде. Угроза — это потенциальная возможность определённым образом нарушить защиту информации. Попытка реализации угрозы называется атакой, а тот, кто предпринимает такую попытку, — злоумышленником (malicious). В то время как информационная безопасность — это состояние защищённости информационной среды, защита информации представляет собой деятельность по предотвращению опасностей, то есть процесс, направленный на достижение этого состояния и состоящий в помещении защищаемой собственности в более безопасное место. Основным принципом противодействия угрозам безопасности информации является превентивность принимаемых мер защиты, так как устранение последствий проявления угроз требует значительных финансовых, временных и материальных затрат. Информационная безопасность — социальная проблема, потому что безопасность бессмысленна, пока человек не определит то, что собой представляет опасность. По характеру обеспечения информационной безопасности выделяют: законодательную деятельность; административную (приказы и другие действия руководства организаций, связанных с защищаемыми информационными системами); процедурную (меры безопасности, ориентированные на людей); программно-техническую. Программно-технические методы защиты интеллектуальной собственности являются существенной частью процедуры проектирования на FPGA.

В FPGA объектом защиты является интеллектуальная собственность, воплощённая в конфигурационном файле. Эта собственность создается производителем БИС FPGA и САПР для её программирования, производителями компонентов (IP cores) и разработчиком устройства.

1. Владельцы интеллектуальной собственности

На рынке энергозависимых FPGA доминируют три продавца БИС FPGA и одновременно производители САПР для программирования своих БИС: Altera, Lattice и Xilinx, каждый из которых через 12–18 месяцев представляет на рынок новое семейство FPGA. Все эти продавцы FPGA не являются владельцами производственных мощностей — они только проектируют FPGA, изготовляют БИС другие компании — «кремневые фабрики». Продавцы FPGA имеют две основные проблемы защиты своей интеллектуальной собственности. Во-первых, они должны защитить топологию и технологию самой БИС FPGA от перепроектирования, копирования или изменений. Во-вторых, они должны обеспечить клиентам возможность защитить их интеллектуальную собственность в ходе проектирования и при эксплуатации у конечного пользователя. Продавцы FPGA заинтересованы в интеграции и всемерном расширении использования компонентов от независимых источников, так как это стимулирует рынок и ведёт к увеличению продаж FPGA и САПР. В то же время продавцы FPGA и компонентов рассматривают разработчиков устройства как угрозу своей интеллектуальной

собственности и принимают меры по её защите. Противником всех троих владельцев интеллектуальной собственности является конечный пользователь устройства.

Разработка компонентов, представляющих собой готовые функциональные описания, является значительным рынком в области производства электронных устройств. Покупая и объединяя их в собственный проект, разработчики устройства экономят в стоимости и времени проектирования. Имеются компоненты таких размеров, что могут занять FPGA целиком. Компоненты распространяются как модули на языке описания аппаратуры или в синтезированном виде как сеть логических элементов. Прибыль от продажи компонентов получается из платежей продавцов FPGA, которые закладывают затраты на разработку собственных САПР и компонентов сторонних разработчиков в цену БИС.

Проблемой производителя компонентов является защита против не имеющего лицензию разработчика устройства. Компоненты лицензируются, распространяются и защищаются теми же способами, как и программное обеспечение. Некоторые поставщики зашифровывают свои компоненты и поставляют специальные инструменты разработки, которые их могут обработать. Недостаток такой защиты — сложность интеграции в существующие САПР и ненадёжность: защита держится на программных инструментах, которые могут быть взломаны.

Другой подход состоит в том, чтобы спрятать в компоненте специальный код, так называемую `watermark`, которая может использоваться для проверки источника собственности. Предложено много подходов к осуществлению инкапсуляции `watermark` в компонент. Большинство из них неприменимо из-за отсутствия реальной возможности проверки `watermark`, которая должна основываться только на исследовании устройств, купленных конечным пользователем. Если от обвиняемого разработчика требуется дополнительная информация, проверка не будет юридически значимой.

Атака на устройство с FPGA выполняется не только для кражи интеллектуальной собственности, воплощённой в функциональных возможностях устройства, но и для нанесения ущерба легальному пользователю устройства. Игнорируя последний случай, можно считать, что противником владельцев собственности является пользователь (недобросовестный). В энергозависимых FPGA коммуникация между FPGA и памятью может быть записана, что даёт возможность перехватить конфигурационный файл. Таким образом, из всех видов атак [1] на цифровые СБИС для FPGA наиболее простой оказывается атака на основе конфигурационного файла. Существующие в настоящее время методы защиты конфигурационного файла [1] имеют тот главный недостаток, что ведут к росту энергопотребления и снижению надёжности и скорости работы. Распространена ситуация, когда защита от перехвата конфигурационного файла не оправдывает затраты.

Для защиты информационных ресурсов широко используются технические средства защиты авторских прав (DRM — Digital Rights Management), которые затрудняют создание копий защищаемых произведений, распространяемых в электронной форме. Обычно в качестве технических средств защиты информации используется её кодирование или шифрование. Хотя эти технические средства призваны воспрепятствовать лишь неправомерному копированию произведений, они не допускают либо ограничивают любое копирование, в том числе добросовестное, поскольку невозможно техническими средствами отличить «законное» копирование от «незаконного». Задача разрешить воспроизведение и в то же время запретить копирование представляет собой принципиально неразрешимую проблему, так как если возможно воспроизведение информации, то возможно и её последующее копирование. В процессе проектирования

FPGA кодирование информации получается «даром», если формат результата проектирования является секретом.

Сам по себе конфигурационный файл не представляет ценности для взломщика, но, декомпилируя его в процессе перепроектирования, можно получить представление исходного алгоритма функционирования, пригодное для изменения и переработки. Предотвращение перепроектирования и представляет собой основную линию обороны владельцев интеллектуальной собственности. Известно изречение: «there is no security in obscurity» («упрятывание — не защита»). Однако скрытие информации остаётся действенным способом защиты интеллектуальной собственности в области проектирования на FPGA. В настоящее время все продавцы FPGA держат формат конфигурационного файла своих устройств в секрете. Кроме того, лицензионное соглашение САПР проектирования FPGA прямо запрещает любое их использование для перепроектирования.

2. Обратное проектирование

Процесс обратного проектирования является существенной частью создания конкурентоспособной продукции и обычно служит средством разработки устройств, более эффективных, чем имеющиеся у конкурентов. Побочная область применения обратного проектирования — перепроектирование на современной базе устарелых компонентов, находящихся в составе долговечного оборудования, такого, как военные или космические системы, ядерные реакторы, авиалайнеры и морские суда.

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 (текст описания поведения устройства — исходные данные для САПР) — описание «промежуточного уровня»; детализированное структурное описание (сеть элементов) для машинного потребления, но, возможно, постижимое людьми — описание «низкого уровня»; продукт (конфигурация настройки FPGA), строение которого обычно не воспринимается невооруженными органами чувств человека, — самый низкий уровень.

Обратное проектирование в общем случае состоит из следующих стадий: анализа продукта; извлечения описания продукта промежуточного уровня; анализа описания продукта интеллектом человека для построения новой спецификации; разработки нового продукта с использованием построенной спецификации. Обратное проектирование является инверсией обычной разработки в смысле порядка процесса преобразований; его задача заключается в построении спецификации на основе анализа продукта. Результат обратного проектирования не гарантирован, в общем случае невозможно даже в теории построить оригинальную спецификацию, изучая только продукт.

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...». Подобные законодательства имеют Япония, Европейский союз и другие страны [2].

Изготовители FPGA не разглашают кодирование `bitstream`, хотя они обеспечивают разработчиков инструментами для их обработки; значительное количество инфор-

мации о bitstream содержится в различных документах. Несмотря на актуальность этой информации, сообщения об успешном обратном проектировании FPGA bitstream отсутствуют. Хотя в 1990-х годах появилось сообщение [3] о раскрытии секрета кодирования bitstream Xilinx в компании «NeoCad», однако, согласно сообщению её руководителей, было декомпилировано программное обеспечение Xilinx для построения bitstream, само кодирование bitstream осталось нераскрытым [4]. Основываясь на сложности задачи обратного проектирования bitstream, большинство разработчиков устройств на FPGA, как и компьютерные программисты, игнорируют риск кражи интеллектуальной собственности путём перепроектирования. Задача декомпиляции bitstream, извлечённого из энергонезависимой памяти в готовом устройстве, безнадёжна. Но если перепроектирование выполняется организацией — разработчиком оригинального устройства на FPGA (например, с целью использования более дешёвой в массовом производстве технологии изготовления), то задача декомпиляции в некоторых случаях может быть решена.

3. Защита САПР

Без программных инструментов автоматизированного проектирования и заложенных в них алгоритмов проектирование FPGA невозможно. Эта интеллектуальная собственность нуждается в особых технических средствах защиты авторских прав. Задачу защиты инструментов САПР техническими средствами можно поставить следующим образом. Дана проблема P и её решение S, нужно определить, получена ли S заданным программным инструментом или алгоритмом. Цель защиты интеллектуальной собственности, воплощённой в структуре устройства, состоит в том, чтобы обеспечить механизм, позволяющий проверить, с определённой степенью доверия, является ли подозрительная часть этой интеллектуальной собственности дубликатом (частичным или полным) другой интеллектуальной собственности. В защите инструментов САПР цель состоит в том, чтобы обеспечить возможность проследить использование этой САПР. Другими словами, нужно определить, использовались ли определённые инструменты САПР (или алгоритмы) при проектировании подозрительного устройства.

Технические средства защиты САПР, предотвращающие использование её инструментов с нарушением лицензионного соглашения, трудно осуществимы и практически не применяются из-за небольшого (в сравнении с рынком компиляторов языков программирования) числа покупателей САПР FPGA. Экономически оправдано использование технических средств обнаружения использования инструментов САПР при проектировании конкретного устройства. Для этого в результат проектирования нужно встроить watermark, подтверждающую использование инструментов защищаемой САПР. Сам формат bitstream служит гарантией его получения средствами защищаемой САПР, и в случае, когда целевой платформой является именно FPGA, использование таких watermark излишне. Для защиты от перепроектирования watermark должна сохраняться после декомпиляции bitstream. С другой стороны, сам процесс проектирования сложен, и есть ситуации, когда невозможно техническими средствами отличить «законное» использование инструмента САПР от «незаконного».

Декомпиляция конфигурационного файла требуется в ходе отладки устройства на FPGA. Отладка устройств нацелена на устранение ошибок проектирования и состоит в выполнении тестов. Тесты разрабатываются таким образом, чтобы продемонстрировать работоспособность устройства. В свою очередь, устройство работоспособно только при вполне определённых условиях. Проектировщик не в состоянии заранее предусмотреть все необходимые для исправной работы условия. Одной из задач тестирова-

ния является выявление этих условий. Окончательный вывод о необходимых условиях работоспособности реального устройства обычно даёт натурный эксперимент. Однако из-за сложности современных FPGA натурные эксперименты очень дороги и приходится обходиться моделированием. Для моделирования поведения FPGA с загруженным конфигурационным файлом необходимо построение модели из примитивов FPGA, для удобства она строится на языке описания аппаратуры. Построение этой модели по сути представляет собой декомпиляцию.

Например, в САПР Xilinx [5] конфигурационный файл является одним из представлений базы данных проекта (файл NCD), содержащим алгоритм функционирования в форме размещённых примитивов FPGA с трассированными соединениями, и строится программой `bitgen`. Программой `netgen` можно построить представление базы данных на языке VHDL, являющейся декомпиляцией конфигурационного файла. Исправления, возникающие в ходе отладки, можно выполнять, изменяя базу данных преобразованием в ASCII-формат XDL [6] и назад.

4. Проблема декомпиляции

В криптографии односторонняя функция — это эффективно вычислимая функция, для задачи инвертирования которой не существует эффективных алгоритмов [7]. Под инвертированием понимается массовая задача нахождения по заданному значению функции одного (любого) значения из прообраза (заметим, что обратная функция, вообще говоря, может не существовать). Декомпиляция конфигурационного файла является обратной функции компиляции конфигурационного файла. Если для функции компиляции конфигурационного файла по заданному исходному представлению (скажем, на языке VHDL) не существует обратной функции, то таким преобразованием обеспечена абсолютная защита конфигурационного файла. Аналитическое представление функции компиляции сложно (фактически это программа компилятора), поэтому теоретическая разработка обратной функции неосуществима. Возможность построения этой функции можно оценить по аналогии с декомпиляцией программ и методом проб и ошибок.

В литературе нет сведений, что проблема декомпиляции конфигурационного файла кем-либо была решена. С другой стороны, известно [1], что информацию о настройке LUT можно извлечь непосредственно из конфигурационного файла. Исследование проблемы декомпиляции программ показало [8], что в общем случае декомпиляция невозможна, потому что декомпиляция самомодифицирующихся программ представляет собой алгоритмически неразрешимую проблему. Относительно декомпиляции конфигурационного файла такого типа результаты в литературе неизвестны.

Эксперименты по декомпиляции представления конфигурационного файла в формате XDL в представление на VHDL показывают, что декомпиляция для важных частных случаев — когда исходное представление на VHDL задаёт комбинационную схему — осуществима практически.

5. Динамическая частичная реконфигурация

Частичная реконфигурация (Partial Reconfiguration — PR) — это возможность реконфигурировать (повторно запрограммировать) часть FPGA, в то время как остальная её часть остаётся неизменной и продолжает функционировать. При частичной реконфигурации в FPGA загружается только часть `bitstream`, а не весь `bitstream` целиком. Динамической эта частичная реконфигурация называется потому, что позволяет перепрограммировать часть FPGA «на лету», в процессе нормального функ-

ционирования, не затрагивая остальной части системы. В противоположность этому статическая реконфигурация состоит в перепрограммировании FPGA, когда она находится в состоянии сброса [9]. В FPGA Xilinx допускается частичная реконфигурация устройств, начиная от Spartan-3 и всех последующих FPGA.

САПР FPGA Xilinx имеет два основных метода частичной реконфигурации, называемых модульной (module-based) и дифференциальной реконфигурацией (difference-based). Развитые средства автоматизации проектирования имеются только для модульной реконфигурации [10]. При модульной реконфигурации в сетке элементов FPGA выделяются области (PR region), которые могут быть перепрограммированы, а само устройство описывается в виде набора модулей, причём для каждой области перепрограммирования указывается набор оверлейных модулей (PR module), которые будут сменять друг друга при перепрограммировании этой области. Каждый оверлейный модуль размещается и трассируется в расчёте на помещение в указанную область; `bitstream` оверлейных модулей записывается в конфигурационный файл вслед за основным `bitstream`. Эти `bitstream` используются в процессе динамической реконфигурации по одному для каждой перепрограммируемой области FPGA.

Размер полного `bitstream` для Spartan-3 порядка 0,5 Мб, поэтому временные затраты на статическую реконфигурацию значительны, так как этот объём данных нужно прочитать из энергонезависимой памяти. Размер `bitstream` оверлейных модулей пропорционален числу конфигурируемых ресурсов — размеру области перепрограммирования, поэтому время, затрачиваемое на динамическую реконфигурацию, значительно меньше времени статической реконфигурации.

При статической реконфигурации соединения между модулями размещаются так же, как и другие соединения. При динамической реконфигурации соединения оверлейных модулей имеют ограничения, они должны использовать те же самые физические провода. Это обеспечивается шинным макросом (bus macro). Шинный макрос в сетке элементов FPGA — это область трассированных соединений, обеспечивающая всем оверлейным модулям одни и те же каналы коммуникации. Реализация шинного макроса обеспечивается САПР Xilinx для каждого типа FPGA.

Частичная реконфигурация осуществляется под управлением внешнего процессора через интерфейс JTAG, когда FPGA находится в состоянии сброса, или под внутренним управлением с помощью специальной логики или встроенного процессора, например PowerPC в FPGA типа Virtex II Pro. Самореконфигурация, то есть частичное перепрограммирование посредством внутренней схемы, устраняет потребность во внешнем процессоре. Самореконфигурация в Xilinx FPGA обычно требует использования интерфейса ICAP (Internal Configuration Access Port). Этот интерфейс обеспечивается встроенным в FPGA процессором или создаётся в FPGA специальным статическим модулем. Этот статический модуль является реализацией конечного автомата, который выполняет чтение частичных `bitstream` из энергонезависимой памяти и передачу данных через интерфейс ICAP, устраняя нужду в процессоре. Частичный `bitstream` содержит всю необходимую информацию о реконфигурации заданной области FPGA. То, что формат `bitstream` является секретом, усложняет аппаратуру самореконфигурации и процесс проектирования динамически реконфигурируемого устройства.

Перепроектирование устройства на FPGA с частичной динамической реконфигурацией требует выделения контроллера самореконфигурации и его удаления, модификации поведения статических модулей, модификации оверлейных модулей в статические и изменения функционирования шинного макроса. Значительную часть этой работы трудно выполнить автоматически. Такие устройства похожи на самомодифицирующи-

еся программы, следовательно, в общем случае декомпиляция `bitstream` принципиально невозможна.

Заключение

Область задач охраны интеллектуальной собственности при проектировании устройств на FPGA очень широка и включает технологические и архитектурные аспекты защиты, уязвимости, специфические для FPGA, ограничения возможностей устройств на FPGA в отношении защиты, проблемы создания технических средств защиты устройств на FPGA и САПР для их проектирования.

Доминирующим подходом для защиты интеллектуальной собственности, воплощённой в FPGA, является методика «водяных знаков» — `watermark`. Предложены методики для включения `watermark` в описания на всех уровнях абстракции; в большинстве из них используются побочные каналы. Побочные каналы исследования устройства основаны на использовании информации о физических процессах в устройстве: задержке, энергопотреблении, электромагнитном излучении, шуме основания, температуре. Эти методики не подходят для создания `watermark`, сохраняющихся при перепроектировании. Создание водяных знаков является намного более трудной задачей для FPGA, чем для интегральных схем специального назначения.

Создателями интеллектуальной собственности при проектировании FPGA являются разработчики программных инструментов автоматизированного проектирования и заложенных в них алгоритмов. Самый радикальный способ защиты САПР — это дистанционное управление, когда продавец САПР может удалённо управлять тем, какое действие может или не может выполнить пользователь. Однако этот метод пока не нашёл широкого распространения.

Сложность задачи обратного проектирования для FPGA признаётся одним из преимуществ FPGA по сравнению с другими реализациями с точки зрения защиты интеллектуальной собственности [11]. Задача декомпиляции `bitstream` в общем случае принципиально неразрешима. Однако эксперименты по декомпиляции представления конфигурационного файла в формате XDL в представление на VHDL показывают, что декомпиляция для важного частного случая — когда исходное представление на VHDL задает схему из триггеров и логических элементов — осуществима практически.

ЛИТЕРАТУРА

1. *Wollinger T., Guarjardo J., and Paar C.* Security on FPGAs: State-of-the-Art implementations and attacks // ACM Trans. Embedded Comput. Systems. 2004. V. 3. No. 3. P. 534–574.
2. *Musker D. C.* Reverse engineering // Protecting & Exploiting Intellectual Property in Electronics, IBC Conferences, 10 June 1998. `www.jenkins-ip.com/serv/serv_6.htm`
3. *Wollinger T. and Parr C.* How secure are FPGAs in cryptographic applications // LNCS. 2003. V. 2887. P. 91–100.
4. *Trimberger S.* Trusted design in FPGAs // Proc. 44th Annual Design Automation Conf. (DAC'07). N. Y.: ACM, 2007. P. 5–8.
5. *Note J.-B. and Rannaud E.* From the bitstream to the netlist // Proc. 16th Int. ACM/SIGDA Symp. on FPGA. N. Y.: ACM, 2008. P. 264–264.
6. *Beckhoff C., Koch D., and Torresen J.* The Xilinx Design Language (XDL): Tutorial and use cases // 6th Int. Workshop ReCoSoC'11, Montpellier, France, 2011. P. 1–8.
7. Введение в криптографию / под ред. В. В. Ященко. СПб.: Питер, 2001. 288 с.
8. *Cifuentes C. and Gough K. J.* Decompilation of binary programs // Software — Practice & Experience. 1995. V. 25. No. 7. P. 811–829.

9. *Beckhoff C., Koch D., and Torresen J.* Go ahead: a partial reconfiguration framework // 20th Annual IEEE Int. Symp. on Field-Programmable Custom Comp. Machines, April 29 – May 1, 2012, Toronto, Canada. P. 27–44.
10. Partial Reconfiguration User Guide UG702 (v13.1). Xilinx Inc., March 1, 2011. 124 p.
11. *Majzoobi M., Koushanfar F., and Potkonjak M.* FPGA-oriented security // Introduction to Hardware Security and Trust / eds. M. Tehranipour and M. Wang. Springer, 2011. P. 195–231.