Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Кафедра ПОИТ

Тема реферата:

«Методы защиты проектных HDL-описания от несанкционированного использования. Применение водяных знаков и отпечатков пальцев (watermarking & fingerprint)»

Выполнил: Верховцов Павел Андреевич

магистрант кафедры ПОИТ

группа № 7М2221

Проверил: Иванюк Александр Александрович

доктор технических наук

Минск 2018

Содержание

[Введение 3](#_Toc510636465)

[1. Методы защиты проектных HDL-описания от несанкционированного использования 4](#_Toc510636466)

[1.1 Обфускация 4](#_Toc510636467)

[1.2 Аутентификация и идентификация цифровых устройств 5](#_Toc510636468)

[2. Применение водяных знаков и отпечатков пальцев (watermarking & fingerprint) 7](#_Toc510636469)

[2.1 Цифровой водяной знак 7](#_Toc510636470)

[2.2 Применение примитива LUT4 для внедрения водяного знака 9](#_Toc510636471)

[2.3 Применение генераторов константных значений 11](#_Toc510636472)

[2.4 Применение конечных автоматов для внедрения водяного знака 13](#_Toc510636473)

[Заключение 14](#_Toc510636474)

[Список использованных источников 15](#_Toc510636475)

# Введение

Сегодня в мире наступил информационный этап развития. Всеобщая компьютеризация дала возможность использовать цифровые технологии во многих сферах жизни. За каждый разработкой, делающей современную жизнь проще и безопаснее, стоит инновационная идея. На этом фоне вопросы обеспечения защиты авторского право и интеллектуальной собственности актуальны как никогда ранее, ведь нарушение авторского права уменьшает реальные доходы инноваторов и инвесторов, а следовательно бюджеты будущих исследований вплоть до полного прекращения исследовательской деятельности. Данные процессы приводят к замедлению технологического прогресса.

Существующие юридические нормы, такие как патенты, коммерческая тайна, авторские права и соглашения о неразглашении сдерживают потенциальных нарушителей. Одного существует проблемой является контроль соблюдения этих норм. Поэтому необходима разработка и внедрение механизмов, позволяющих обеспечить возможности эффективного обнаружения и последующего отслеживания случаев нарушения авторских прав.

# Методы защиты проектных HDL-описания от несанкционированного использования

Средства защиты разрабатываются и внедряются не на пустом месте основным двигателем являются новые и усовершенствованные вектора атак на цифровые системы.

Выделяются следующие типы атак:

1)Read Back – чтение бит-образа с целью анализа.

2)Cloning (Bit cloning) –тиражирование системы в обход соглашения с производителями для изготовления более дешевой цифровой системы ПЛИС и использование BIT образа для её конфигурации.

3)Reverse engineering – применяем обратное проектирование на бит-образ и получаем схему. Восстановление RTL и нахождение не стандартных участков в схеме.

4)Несанкционированная модификация, передача либо использование VHDL описания.

Для защиты интеллектуальной собственности и предотвращения вышеупомянутых атак используются следующие методы защиты:

* Запутывающие преобразования (obfuscation)
* “Водяные знаки” разработчика (watermarking)
* “Отпечатки пальцев” пользователей (fingerprinting)
* Привязка к цифровой системе (авторизация и идентификация)
* Шифрование .BIT-образа

## 1.1 Обфускация

Обфускация или запутывание кода — приведение исходного текста или исполняемого кода программы к виду, сохраняющему ее функциональность, но затрудняющему анализ, понимание алгоритмов работы и модификацию при декомпиляции [1].

Методы обфускации подразделяются на лексический и функциональные. Лексическая обфускация заключается в форматировании описания, изменении его структуры, таким образом, чтобы оно стал нечитабельным, менее информативным, и трудным для изучения. К методам лексической обфускации относится:

* Удаление комментариев
* Удаление символов разделителей
* Замена пользовательских идентификаторов

Основным недостатком лексической обфускации является идентичность результатов синтеза обфусцированного и не обфусцированного описания [2].

Пример:

HDL – описание до обфускации:

process (cwait, start, rom\_data)

begin

if (start = '1' and c\_adr = "0000") then

n\_adr <= "0001";

elsif (cwait = '1' and c\_adr = "0001") then

n\_adr <= "0000";

else

n\_adr <= rom\_data(6 downto 3);

end if;

end process;

HDL-описание после обфускации:

process (qw,hrttrrg,bgede) begin if(hrttrrg = '1' and antfr = "0000") then nrfee <= "0001";elsif (qw = '1' and antfr = "0001") then nrfee <= "0000";else nrfee <= bgede(6 downto 3);end if;end process;

Функциональная (схемная) обфускация заключается в изменение синтезируемой схемы без изменения логики работы. Основными приемами при функциональной обфускации являются соединение комбинационной и последовательной схем во избежание оптимизации (запутывание) синтезатора, использование генераторов константных значение на основе SLI (simple login implication). Однако SLI может быть обнаружен современными синтезаторами что делает ее использование малоэффективным.

## 1.2 Аутентификация и идентификация цифровых устройств

Идентификация и аутентификация — это первая линия обороны, «проходная» информационного пространства организации.

Идентификация позволяет субъекту (пользователю, процессу, действующему от имени определенного пользователя, или иному аппаратно-программному компоненту) назвать себя (сообщить свое имя). Посредством аутентификации вторая сторона убеждается, что субъект действительно тот, за кого он себя выдает. В качестве синонима слова «аутентификация» иногда используют словосочетание «проверка подлинности».

Для цифровых систем под аутентификацией часто понимают проверку «правильности окружения».

Подходы используемы для аутентификации и идентификации можно разделить на:

1. Аппаратные (на уровне FPGA):
   * + Использование криптопроцессоров;
     + Шифрование .bit образа.
2. На уровне системы.

Стоит отметить что несмотря на высокую надежность аппаратные способы целесообразно применять только в дорогостоящих устройствах.

# Применение водяных знаков и отпечатков пальцев (watermarking & fingerprint)

Водяные знаки и отпечатки пальцев с технической точки зрения представляют собой эквивалентные процессы внедрение секретной информации в HDL-описание. Однако они имеют принципиально разное предназначение, так водяные знаки служат для доказательства авторства, а отпечатки пальцем для идентификации пользователя(покупателя) продукта. Отпечаток пальца позволяет производителю отслеживать соблюдение условий использования своего продукта и выявлять покупателей, занимающихся незаконным тиражированием.

## 2.1 Цифровой водяной знак

Под водяным знаком понимают внедренный специальный уникальный идентификатор автора для возможности в дальнейшем извлечь этот водяной знак и доказать, что автором этого водяного знака, а также метода его внедрения является разработчик и никто другой.

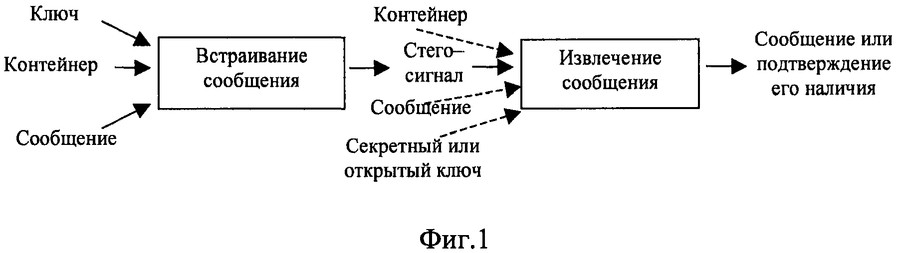


Рисунок 2.1 – Жизненный цикл цифровых водяных знаков

Защита на основе водяных знаков состоит из двух этапов:

* синтез (встраивание)
* обнаружение (извлечение)

Жизненный цикл ЦВЗ, представленный на рисунке 2.1, может быть описан следующим образом:

1. Сообщение шифруется с помощью ключа (секретного в случае асинхронного шифрования).
2. Полученный водяной знак внедряется с помощью заданного алгоритма в контейнер.
3. Извлечение водяного знака из контейнера.
4. Дешифровка водяного знака с помощью ключа (открытого в случае асинхронного шифрования)
5. Сравнение полученного сообщения с оригинальным.

Алгоритм используемый для внедрения и извлечения водяного знака, не обязательно быть проприетарным, в то время как код, используемый в процессе шифрования, должен быть секретным для любого выпущенного продукта [3].

Основные требования к водяному знаку:

* прозрачность (не нарушать функциональности устройства),
* прочность (трудность удаления или подделки),
* извлекаемость (легкость извлечения).

ЦВЗ могут быть трех типов:

* робастные
* хрупкие
* полухрупкие (semifragile).

Под робастностью понимается устойчивость ЦВЗ к воздействиям различного рода [4].

Хрупкие ЦВЗ разрушаются при незначительной модификации заполненного контейнера. Они применяются для аутентификации сигналов. Также отметим то, что хрупкие ЦВЗ должны не только отразить факт модификации контейнера, но также вид и местоположение этого изменения.

Полухрупки е ЦВЗ устойчивы по отношению к одним воздействиям и неустойчивы по отношению к другим. Вообще говоря, все ЦВЗ могут быть отнесены к этому типу. Однако полухрупкие ЦВЗ специально проектируются так, чтобы быть неустойчивыми по отношению к определенного рода операциям. Например, они могут позволять выполнять сжатие контейнера, но запрещать вырезку из него или вставку в него фрагмента.

Так что стоит обеспечить что извлечение водяного знака приводило к нарушению функционирования или требовало переработки основного ядра. Внедрение своего водяного знака могло нарушить функционирование и повлиять на производительность.

Водяные знаки следует выполнять одновременно на различных уровни абстракции для того, чтобы повысить надежность подхода и обеспечить быстрое и точное отслеживание последнего лицензиата, который в конечном итоге вызвал нарушение.

Выделяются следующие области внедрения водяных знаков в цифровые системы [2]:

* поведенческое описание (HDL);
* структурный уровень;
* физическая реализация.

## 2.2 Применение примитива LUT4 для внедрения водяного знака

Принцип использования LUT (look-up table) для внедрения водяного знака заключается в подключении одного или нескольких входов элемента к источникам константных значений [5].

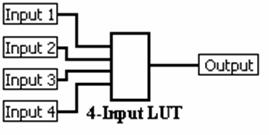


Рисунок 2.2 – Примитив LUT4

LUT4

generic map (

INIT => ‘0110100100100001’)

Port map (

O => Q,

Inp0 => A,

Inp1 => B,

Inp2 => ‘0’,

Inp3 => D

);

Таблица 2.1 – Соотношение входных и выходных сигналов LUT4 с внедренным водяным знаком

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inp1 | Inp2 | Inp3 | Inp4 | Out |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |

Продолжение таблицы 2.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inp1 | Inp2 | Inp3 | Inp4 | Out |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

Строки, отмеченные красным цветом, не используются при работе в штатном режиме и служат контейнером для водяного знака. При необходимости можно увеличить объем внедряемой информации можно использовать несколько элементов.

При использовании приведённой схемы единственным способ извлечения водяного знака является обратное проектирование что не всегда возможно. Для решения этой проблемы может быть использована схема мультиплицирующая константные значения на вход LUT в зависимости от входного сигнала, рисунок 2.3. При подаче на вход WME водянок знак может быть извлечен без применения средств обратного проектирования.

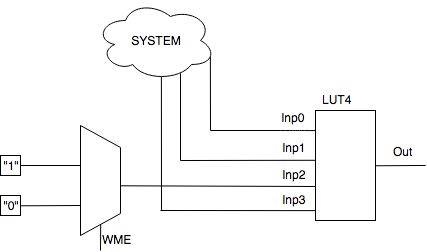


Рисунок 2.3 – Схема извлечения водяного знака с мультиплексором

Еще одним способом использование LUT является переведение их в режим сдвигового регистра (Shift register) поддерживаемый рядом ПЛИС. Несколько LUT элементов могут быть объединены в каскадную схему и по сигналу сдвига начинается смещение битов по всему каскаду с последующей проверкой значений битов на заранее определенных позициях. Основной недостаток данного подхода является необходимость восстановления исходной конфигурации после проведения процедуры выделения водяного знака. Также применение данного подходя ограничено моделями ПЛИС поддерживающими режим сдвигового регистра [6].

Также стоит отметить что обычно проект содержит большое количество не полностью используемых LUT-элементов, вследствие особенностей алгоритмов синтезатора и структурных особенностей ПЛИС [6]. Данных факт дает возможность внедрять достаточный объем информации в целях обеспечения защиты интеллектуальной собственности, в частности «водяные знаки» и «отпечатки пальцев».

## 2.3 Применение генераторов константных значений

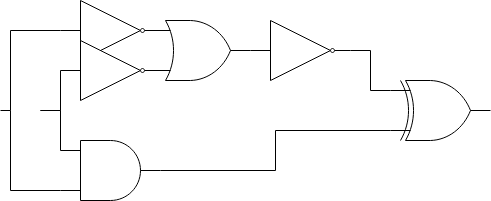
Идея данного подхода состоит во внедрении последовательности элементов генерирующих водяной знак.

Генератором константного значения называется схема результат работы, которой (константный) известен в процессе проектирования, но сложен для определения в процессе анализа [6].

Для генерации константных значений можно использовать комбинационные схемы с заранее известным выходным значением, например, для генерации нуля можно использовать схему описываемому следующим выражением Q = (a∧b)⊕¬(¬a∨¬b) при всех наборов входных сигналов выходной будет равен нулю.

Таблица 2.2 – Таблица истинности генератора константного нуля

|  |  |  |
| --- | --- | --- |
| a | b | Q |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |



a

b

Рисунок 2.4 – Схема генератора константного нуля

Для получения единицы достаточно инвертировать выходной сигнал.

Также стоит отметить что существует больше количество логических схем подобных приведённой, однако современные синтезаторы могут распознать подобные генераторы, что делает их применение для кодирования водяного знака не целесообразным и рискованным [5].

Альтернативным подходом является использование последовательного домена для схожих целей.

Как пример можно использовать D-триггер, рисунок 2.5:

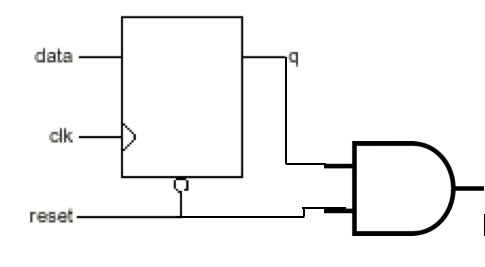


Рисунок 2.5 – Схема генератора константного нуля на основе D-триггера

Для доказательства рассмотрим два примера:

* Reset = 1, D = X

Так как вход reset является асинхронным то предыдущее значение триггера сбрасывается и на выходы q появляется 0, следовательно, результирующий сигнал тоже будет равен 0.

* Reset = 0, D = X

При низком значении на входе reset триггер работает в штатном режиме и на выходе q может быть как 0, так и 1, однако поскольку reset равен нулю, результирующий сигнал тоже будет равен 0.

Таким образов результирующий сигнал всегда будет равен 0, вне зависимости от текущего состояния триггера.

Приведенный пример комбинирует в себе последовательного и комбинационного доменов, что затрудняет его обнаружение синтезатором, поскольку компоненты, отвечающие за последовательный и комбинационный домен в них разнесены.

Также для генерации константных значений могут быть использованы линейные сдвиговые регистры с обратной связью (LFSR). Основываясь на свойстве цикличности генерируемой последовательности с прохождением через все состояния размерности n, за исключением всех нулей. Очевидно, что самым простым вариантом является инициализация всех разрядов нулями, что позволит получить генератор константного нуля. Однако существуют и менее очевидные способы, что увеличивает шанс избежать распознавания и оптимизации при синтезе.

Одним из таких вариантов основан на цикличности состояний регистра и обходе нулевого состояния, благодаря этому можно быть уверенным что, хотя бы один бит в регистре всегда равен единице, соединив выход каждого разряда к логическому ИЛИ получим генератор единицы.

## 2.4 Применение конечных автоматов для внедрения водяного знака

Суть подхода в нахождении неиспользуемых последовательностей входных/выходных сигналов и использовании их как водяной знак. Также возможно расширить размерность входных сигналов для увеличения объема внедряемой информации [6, 7].

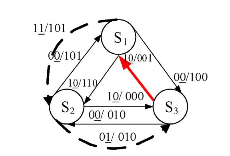
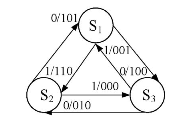


Рисунок 2.6 – Граф до и после внедрения водяного знака

Как показано на рисунке 2.6 размерность входного сигнала была расширена до 2-х бит, при этом работа конечного автомата не была нарушена, при работе в штатном режиме анализируется только первый бит входного сигнала.

При необходимости извлечь водяной знак производится обход графа по неиспользуемым ветвям в определенном порядке и из выходных сигналов собирается значение водяного знака.

# Заключение

Защита авторских прав интеллектуальной собственности является серьезной задачей и проблемой в настоящий момент. Это связано с отраслевым сдвигом, при котором электронные схемы легко доступны в виде виртуальных блоков на любых уровнях абстракции, что позволяет злоупотреблять соглашениями об использовании и красть объекты интеллектуальной собственности.

В данном реферате было описано несколько способов создания водяных знаков на уровнях HDL-описания. Внедрения вышеописанных подходов повысить эффективность обнаружения, отслеживания нарушений интеллектуальной собственности и проведения криминалистического анализа. Также знание о наличии водяного знака или отпечатка пальцев может остановить потенциального злоумышленника, тем самым позволит снизить расходы на потенциальные судебные разбирательства

# Список использованных источников

[1] Chernov, A. New Program Obfuscation Method / A. Chernov // International Conference «Perspectives of Systems Informatics». — 2003. — no. 5.

[2] Collberg, C. Taxonomy of Obfuscating Transformations / C. Collberg, C. Thomborson, D. Low — The University of Auckland Press, 1997.

[3] Tehranipoor, Mohammad. Introduction to Hardware Security and Trust / Mohammad Tehranipoor, Cli Wang. — Springer Publishing Company, 2011. — 436 P.

[4] Грибунин, В. Г. Цифровая стеганография / В. Г. Грибунин, И. Н. Оков, И. В. Туринцев. — СОЛОН-Пресс, 2002.

[5] Lach, John. Robust FPGA Intellectual Property Protection Through Multiple Small Watermarks / John Lach, William H. Mangione-Smith, Miodrag Potkonjak // Design Automation Conference. — 1999. — no. 36.

[6] Sergeichik, Vladimir V. Implementation of opaque predicates for FPGA designs hardware obfuscation / Vladimir V. Sergeichik, Alexander A. Ivaniuk // Journal of Information, Control and Management Systems. — 2014. — no. 2. — Pp. 177–188.

[7] A Robust FSM Watermarking Scheme for IP Protection of Sequential Circuit Design / Aijiao Cui, Chip-Hong Chang, So `ene Tahar, Amr T. Abdel- Hamid // IEEE Transactions on computer-aidee design of Integrated Circuits and Systems. — 2011. — no. 5.

[8] Abdel-Hamid, A.A Survey on IP Watermarking Techniques / A. Abdel-Hamid, Sofiene Tahar, El Mostapha Aboulhamid // Design Automation for Embedded Systems. 2004. Vol. 9. Pp. 211-227.