(Intellectual Property Protection)

Contents

- What is Intellectual Property
- IP Piracy and Overbuilding Threat
- Countermeasures for IP Protection
- IP Piracy Detection
- Digital Watermarking
- Different Watermarking Techniques

Intellectual Property (IP)

- Intellectual property is an original idea which can be used to earn money.
  - **Example: product, technology, software, …**

- **Design IP:** any innovation and technology that makes design better.
  - Design algorithm, technique, methodology
  - Microprocessor, memory, Verilog chip description

- **Types of IPs:**
  - **Hard IP:**
    - Examples: GDSII file with test lists and high-level model, custom physical layout, fully placed and routed netlist for specific technology library
    - Predictable (optimized) performance, not flexible
  - **Soft IP:**
    - Example: synthesizable HDL source code Very flexible, unpredictable performance
  - **Firm IP:**
    - Examples: placed RTL sub-blocks, fully placed netlist for a generic library.

Design Reuse and Design IP

- The ever-increasing logic density has resulted in more transistors on silicon than designer's ability to design them meaningfully. This creates the design productivity gap between what can be built and what can be designed.

- To bridge this gap the Concept of Design reuse was developed. In this method, large previously designed blocks, such as bus controllers, CPUs, and memory subsystems are reused in ASIC architecture to deliver routine functions.

- This makes it possible to have the new products on market in a timely and cost-effective fashion.

What is a Design IP?

- According to the Reuse Methodology Manual, design intellectual property is a design unit that can be reasonably viewed as a stand-alone subcomponent of a complete SOC design.

- Design IPs can be categorized into three groups based on their performance and flexibility.
  - **Hard IPs** include graphical design system II (GDSII) file with test lists and high-level model, custom physical layout, fully placed and routed netlist for a given technology library.

  - **Soft IPs** are delivered in the form of synthesizable hardware description language (HDL) codes like Verilog or VHDL programs. They offer great design flexibility despite the less predictable performance.

  - **Examples of Firm IPs** are placed register transfer level (RTL) sub-blocks and fully placed netlist for a generic technology library.

Need for securing design IP

- To make the design reusable the IP designers make their IP flexible. To facilitate reuse designers are forced to reveal the details.

- These efforts have made **IP piracy and infringement easier than ever**. Since early 1990s, litigation cases in semiconductor sector related to IP infringement have been increasingly rapidly, causing an estimated annual revenue loss of billions of dollars

- One of the most dangerous threats of **IP piracy comes in the form of reverse engineering**.
- **Reverse engineering** is the process where an object is taken apart and analyzed in-order to improve or duplicate the object.

- The design IP is vulnerable to many such threats, therefore there is a need to secure the IP before it is made public for reuse. IP can be secured in many ways.

IP Piracy and IC Overbuilding

- **IP Piracy:** An attacker with access to an IP or an IC can steal and claim ownership.
- **Overbuilding:** Attacker can also overbuild ICs and can sell them illegally in the Gray market.
- In reuse-base design IPs are developed not only for use, but also for reuse.
- Design engineers are forced to cooperate and share their data, expertise, and experience to make reuse more convenient.
- These efforts has made IP piracy and the infringement much easier than ever.

IP Piracy and IC Overbuilding: Threat

- **Threat Model:**
  - In scenario 1, the attacker in the integration house may pirate the 3PIP and/or overbuild.
  - In scenario 2 and 3, the attacker in the foundry may pirate the 3PIP or IC design.

Table: Scenarios for IP Piracy and IC Overbuilding. Obfuscation (O), Watermarking (W), Fingerprinting (F), and Metering (M) Are the Defenses

Scenario	3PIP Vendor	SoC Integrator	Foundry	User
1	O+W+F+M *	•	—	—
2	O+W+F+M *	—	•	—
3	—	O+W+F+M *	•	—

- The Bullet (•) Depicts an Attacker, the Star (*) Represents a Defender, and the Dash (—) indicates an Untrustworthy Entity.

Countermeasures for Piracy and Overbuilding

- **Goal of IP Protection:**
  - Enable IP providers to protect their IPs against unauthorized use
  - Protect all types of design data used to produce and deliver IPs
  - Detect use of IPs
  - Trace of IPs
- **IP Protection State-of-the-Art**
  - Deterrent
  - Protection
  - VSI tagging standards
  - Detection

IP Protection State-of-the-Art

- **Deterrent:**
  - Rather than preventing IP piracy, discourage the misuse of IPs because the attacker was being caught.
  - Example include, patent, contract, legal enforcement, partnership, etc.
- **Protection:**
  - Prevent and authorize the access to the IP
  - **Example:** Logic obfuscation, Logic encryption/Locking, chemicals, and dedicated hardware.
- **VSI tagging standards:**
  - Industry organizations have been working on establishing standards for IP reuse and IP protection.
  - For example, Virtual Socket Interface Alliance has published two tagging standards to protect hard IPs and soft IPs.
- **Detection:**
  - Enable IP providers to identify the occurrence of IP piracy
  - Identification will be the first step towards legal fight against IP infringement
  - **Example:** Digital watermarking, Digital fingerprinting and IC metering

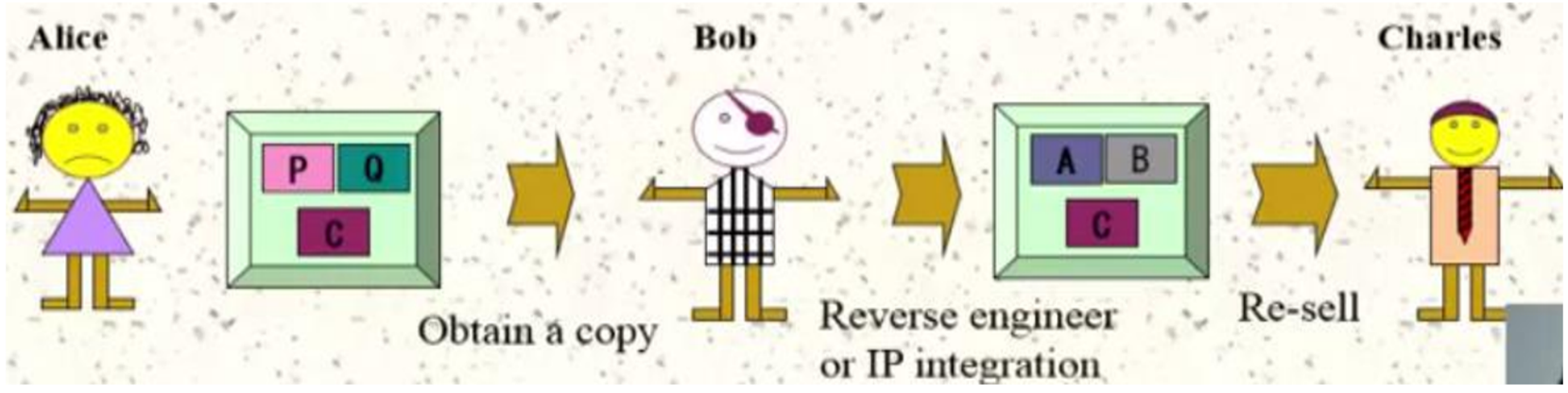
(IP Protection using Watermarking)

IP Piracy Detection: Watermarking

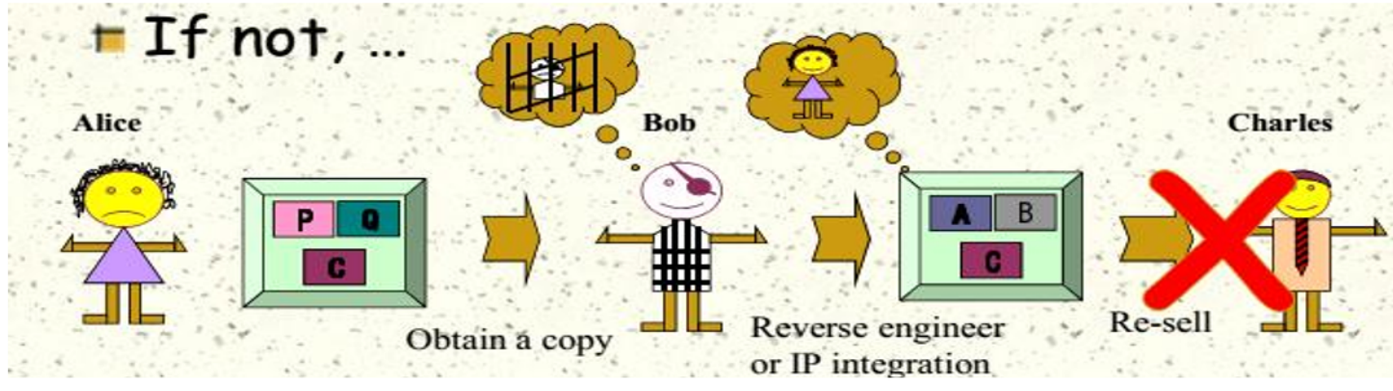
- In watermarking designer's signature is embedded into the design artifact and later designer can reveal the watermark and claim for ownership of an IC/IP.

- Piracy on unprotected IPs

  ❑ When Bob is caught, Alice may not have sufficient evidence to prove her authorship of the IP



- When IPs are watermarked
  - Has Alice's watermark been removed?



- Digital watermarking has been widely used for identification, and notation and copyright of multimedia data such as text, image, audio, and the video.

Watermarking : Basic Idea

- **Challenge:**
  - Traditional watermarking cannot be directly applied for the protection of hardware design IPs, because the value of these IPs rely on their correct functionality, performance, quality, etc.

- **Observation:**
  - For a given system specification, there are normally many different ways to implement the system, or develop the IPs

- **Approach:**
  - If the IP designer can intentionally create certain structures in the implementation of the IP to make it rather unique, then this evidence can be used as watermark to prove IP's authorship.

Watermarking : Requirements

An effective watermark must satisfy the following requirements:

- **High credibility:** Watermark should be readily detectable for the proof of authorship. The probability of coincidence should be low.
- **Low overhead:** Degradation of the software or design by embedding the watermark should be minimized.
- **Resilience:** Watermark should be difficult or impossible to remove without the complete knowledge of the software or design.
- **Transparency:** The addition of the watermark to software and designs should be transparent so that it can be used for existing design tools.
- **Perceptual invisibility:** The watermark must be very difficult to detect. In general, exposing the watermark to the public may make it easier to remove or alter the watermark.
- **Part protection:** Ideally, a good watermark should be distributed all over the software or design in order to protect all parts of it.

Watermarking : Example

- **Watermarking a 4 Bit ALU**

Original design

Same design with message “UMCP TERPS” (in ASCII) embedded.

Watermarking : Example (Cont…)

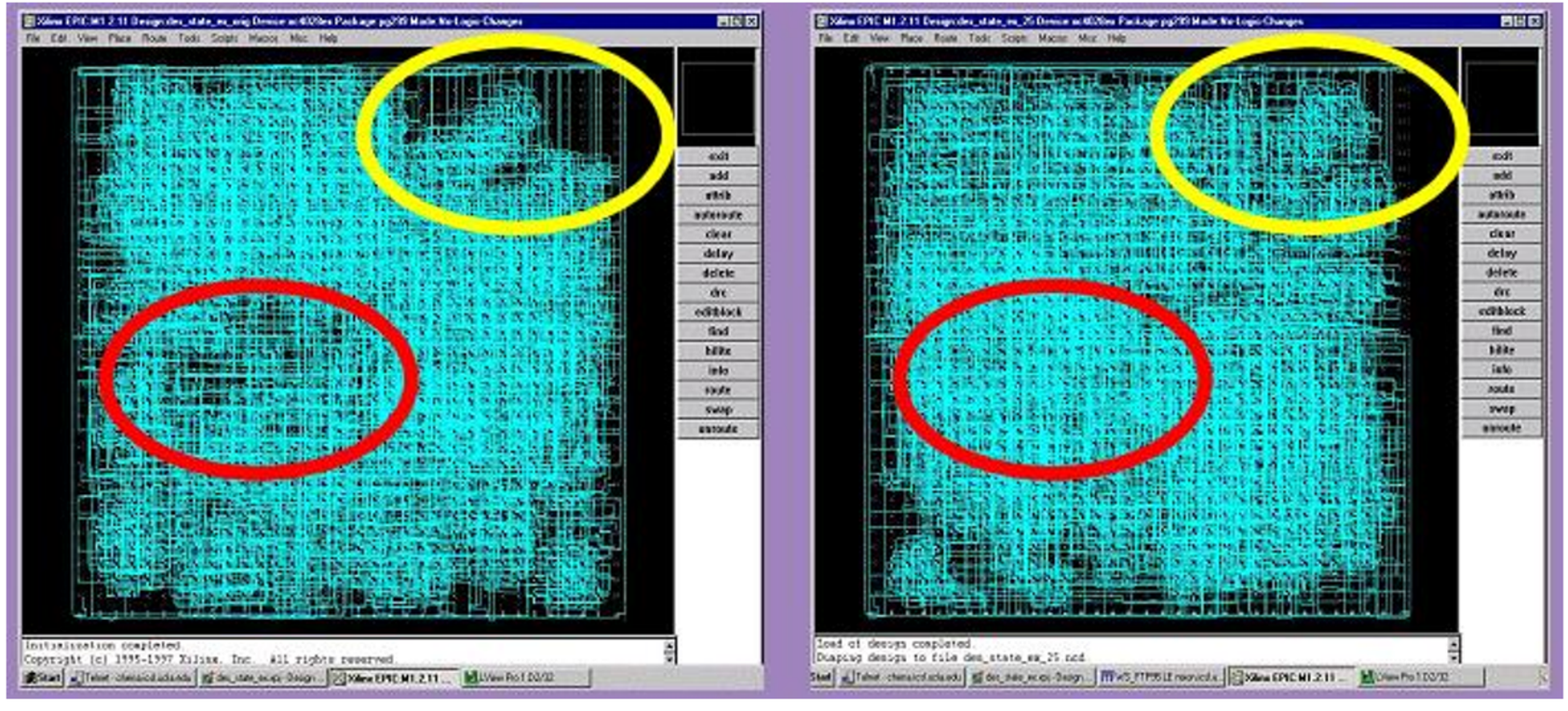
- **Watermarking a 4 Bit ALU**

Original gate-level design

Same design with message "UMCP TERPS" (in ASCII) embedded.

Watermarking : Example

- **Watermarking DES Benchmark**



Same functionality, area, and performance with a 4768-bit watermark embedded in the FPGA design

Watermarking : Watermark Boolean Formula

- **Watermarking a Boolean Formula:**

- **Problem:** Rewrite the following Boolean expression with the minimal number of literals

  F(a,b,c,d) = a'bc'd'+ a'bc'd + a'bcd + abc'd

  don't care conditions: a'b'c'd', abcd

- **Goal: P**rotect the solution (IP)

- **Approach:**
  - Hide one bit with each don't care condition

    make F(a,b,c,d) = 1 to hide a bit '1';

    make F(a,b,c,d) = 0 to hide a bit '0'.

Watermarking of Boolean Formula: Example

- **The original problem:**

  F(a,b,c,d) = a'bc'd' + a'bc'd + a'bcd + abc'd

  Don't care conditions: a'b'c'd', abcd
- **To hide "01"**
  - F = a'b'c'd' + a'bc'd + a'bcd + abc'd + abcd
  - Solution: F = a'bc'+bd
- **To hide "10"**
  - F = a'bc'd' + a'bc'd + a'bcd + abc'd + a'b'c'd'
  - Solution: F = a'c'd' + a'bd + bc'd
- **To hide "00"**
  - F = a'bc'd' + a'bc'd + a'bcd + abc'd
  - Solution: F = a'bc' + a'bd + bc'd
- **To hide "11"**
  - F = a'bc'd'+ a'bc'd + a'bcd + abc'd + a'b'c'd' + abcd
  - Solution: F = a'c'd' + bd

Watermarking of Boolean Formula: Example (Cont...)

- The original problem:

  F(a,b,c,d) = a'bc'd' + a'bc'd + a'bcd + abc'd

  Don't care conditions: a'b'c'd', abcd

- Hide watermark by forcing a '1' or '0' on each don't care condition

- Any 2-bit watermark can be embedded
  - "00": F = a'bc' + a'bd + bc'd
  - "01": F = a'bc' + bd
  - "10": F = a'c'd' + a'bd + bc'd
  - "11": F = a'c'd' + bd

watermark challenge: fairness

Watermarking : Watermark Encoder Design

- **Watermarking an Encoder Design :** A radix-4 to binary encoder

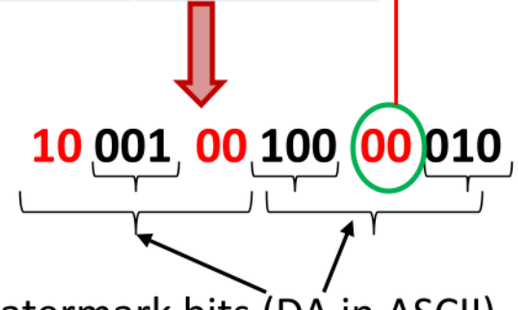
INPUT	OUTPUT
1000	00
0100	01
0010	10
0001	11

(a) Truth table of a radix-4 to binary encoder

extract don't cares

INPUT	OUTPUT
0000	XX
0011	XX
0101	XX->00
0110	XX
0111	XX
1001	XX
1010	XX
1011	XX->00
1100	XX
1101	XX->10
1110	XX
1111	XX

(b) List of (12) don't care inputs



(c) watermark bits (DA in ASCII)

generate new truth table

INPUT	OUTPUT
1000	00
0100	01
0010	10
0001	11
0101	00
1011	00
1101	10

(d) Watermarked truth table of the encoder

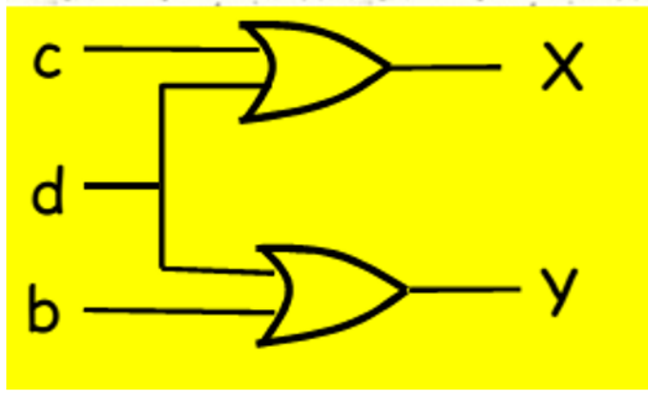
Watermark Encoder Design : Example (Cont…)

- **Watermarking an Encoder Design :** A radix-4 to binary encoder

INPUT	OUTPUT
1000	00
0100	01
0010	10
0001	11

$X = c + d$

$Y = b + d$



INPUT	OUTPUT
1000	00
0100	01
0010	10
0001	11
0101	00
1011	00
1101	10

$X = c + a'b'$

$Y = bc + b'c'd$

watermark challenge: design overhead

There are 8 literal, whereas in above only 4 literal

Summary of the Class

- IP Piracy and Overbuilding
  - Threat and Introduction
  - IP Protection State-of-the-Art
    - Deterrent
    - Protection
    - VSI tagging standards
    - Detection
      - Watermarking
      - **Fingerprinting**
      - **Metering**

Thank You!