SCIENCE
PASSION
TECHNOLOGY

SIP Seminar Topic 14

Reverse Engineering ICs

Martin Mastnak

13.12.2023

1. Introduction

Definition

„Reverse Engineering is the process through which one attempts to understand how a previously made device acomplishes a task with very little to none insight into how exactly it does so.“ [7]

Examples of Reverse Engineering (RE)

- Software
  - Jailbreaking devices to use third-party software
  - Emulators for old game consoles
- RE is quite the norm in the automotive industry [12]
- Extracting Keys from Smartcards [3]
  - In the 2000s: Getting access to satellite television
- Product Teardowns and System Level Analysis [6]

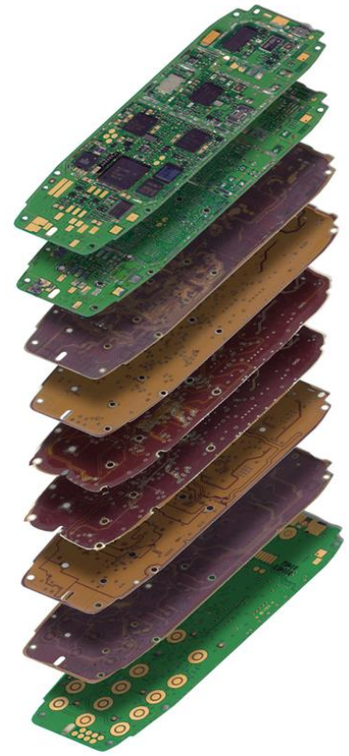


Fig. 1: System Level Analysis [6]

Why Reverse Engineering?

- Analyse HW of competitors to get an overview about industry developments [1] 🧐
- Make sure no modifications have been made (HW Trojans) 🔓
- Replace out-of-service parts of HW 🔧
- Reproduce designs without much development effort 🤑
- Blatantly clone a device without the attempt to understand the design 🦹

2. Fundamentals of IC Reverse Engineering

History

In the 1960s and 70s: [3]

- RE was used by Soviets and Chinese to clone chips
- They could not develop ICs themselves
- Cheaper => cost jobs in Western companies
- At the time RE was widely frowned upon

History

30 years ago: RE was comparatively easy [6]

1. Remove package
2. Put silicon die under optical imaging equipment
3. Put images together
4. Analyse functionality by looking at the pictures ("crawl around-the-floor technique")

History

Fig. 2: How Reverse Engineering ICs was done [6]

Today

- IP providers make sure, that other companies do not use their IP without permission by using RE [3]
- Semiconductor manufacturers want:
  - Get an overview about the competition and its achievements
  - The testing process of ASIC production is in some ways similar to RE
- FPGA design reconstruction is also of concern to developers [4]

Today

- Approaches with lots of manual work not suitable
- Complexity of chips increases due to *Moore's Law*
  - Billions of gates on a complex chip today
- More powerful and automated tools are needed

Legality

- In the USA RE is legitimate according to the *US Semiconductor Chip Protection Act* from 1984 [6]
- RE is always allowed for e*ducational purposes*
- But blatantly copying designs (*trade secrets*) is forbidden
- Similar laws apply in the EU

3. Methods in IC Reverse Engineering

How To IC RE?

1. Netlist Extraction [1]
2. Specification Discovery
   - For analog circuits: Transistor level schematics are copied, including electrical parameters

Other Ideas:

- Social Engineering to get access to the design
- Fault-Insertion to enter debug mode

Netlist Extraction

"Generate a (human-readable) gate-level-netlist description of the chip" [3]

Netlist Extraction

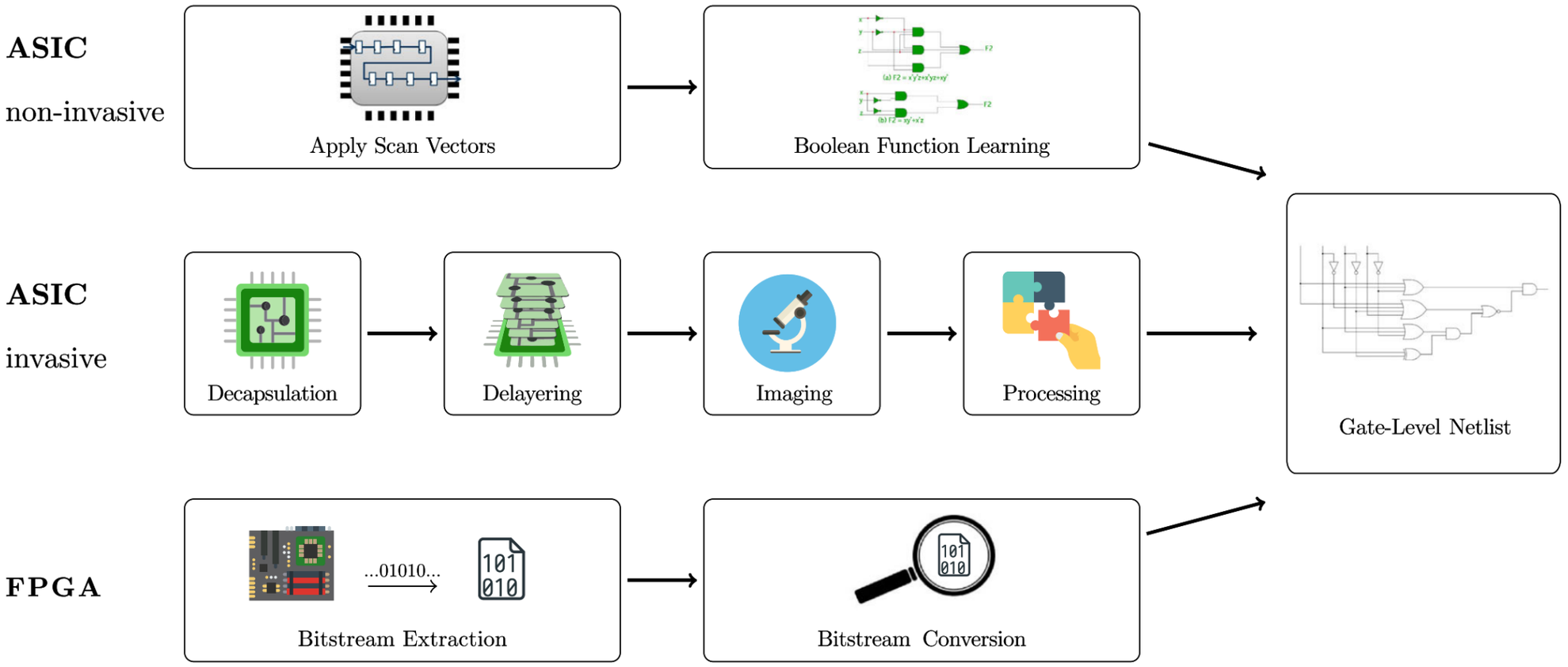


Fig. 3: Types of Netlist Extraction Processes for ASICS and FGPAs [1]

Invasive Netlist Extraction

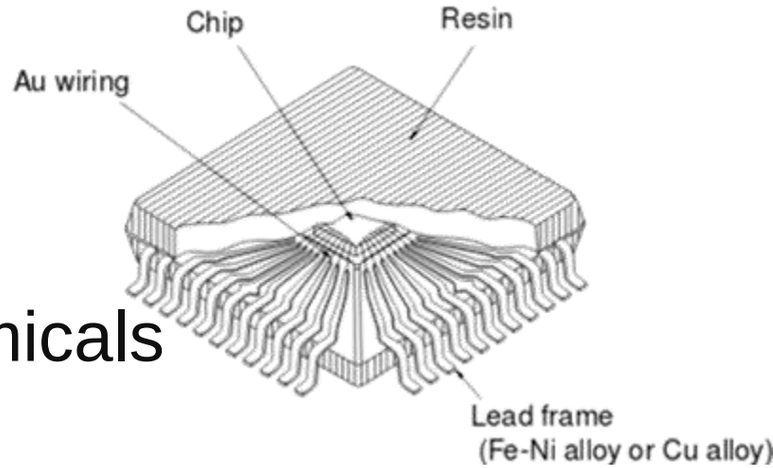


Fig. 4: Die of an IC [11]

1. **Decapsulation:** Packaging is removed with chemicals
2. **Delayering**
   - One layer after the other is removed from the die
   - Strongly dependent on the used technology
   - Many different materials with different characteristics are used [6]

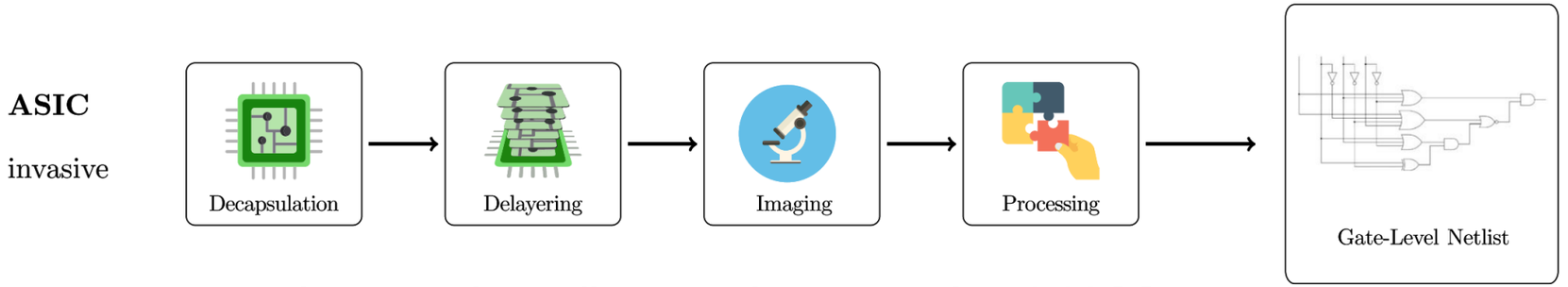


Fig. 5: Invasive Netlist Extraction Process for ASICS [1]

Invasive Netlist Extraction

3. **Imaging:** pictures of each layer are made
   - Optical
   - SEM for technologies with sizes smaller than 1 µm [10]
4. **Processing:** all the images are put together
   - Alignment must be ensured
   - Special software extracts gate-level netlist by identifying standard cells and interconnections in the metal layers

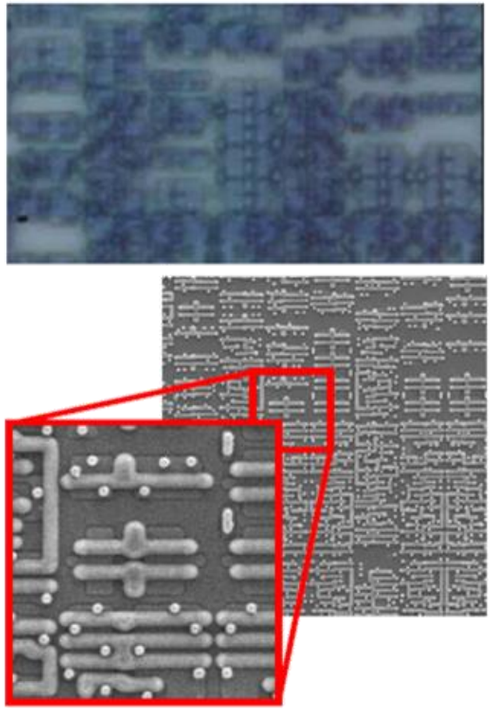


Fig. 6: Optical (top) and SEM image of 130 nm chip [6]

Netlist Extraction in FPGAs

- Bitstream stores basic logic elements and the connection between them
- At every startup bitstream is loaded and codified into the FPGA
- Often stored in SRAM or flash memory

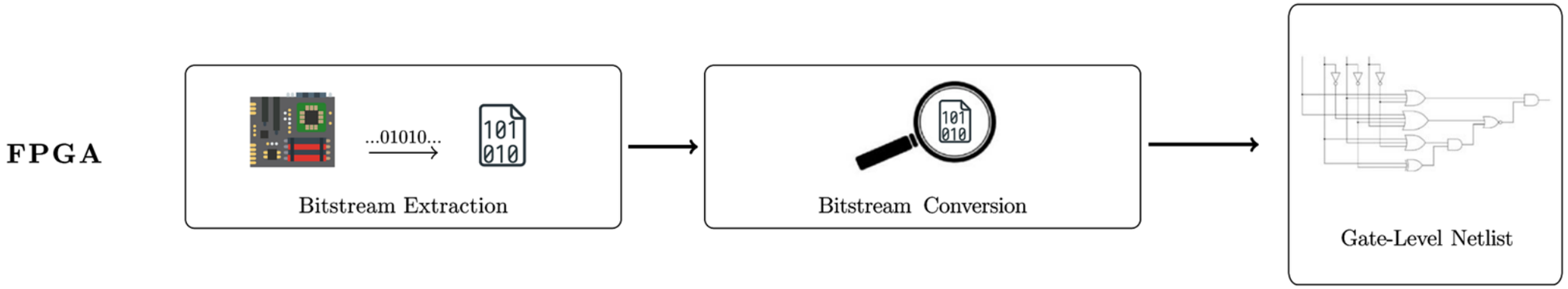


Fig. 7: Netlist Extraction Process for FPGAs [1]

Netlist Extraction in FPGAs

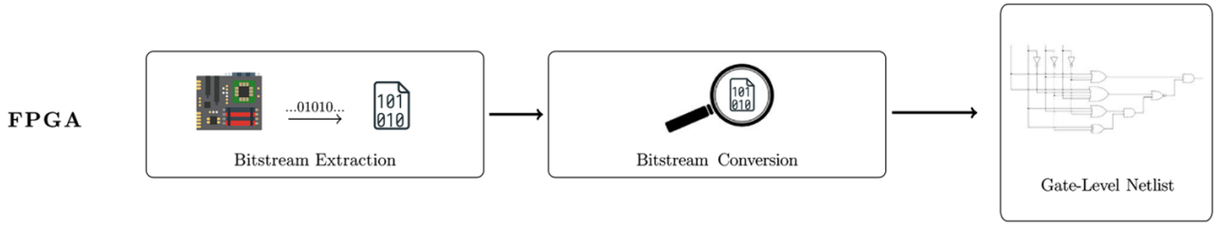


Fig. 8: Netlist Extraction Process for FPGAs [1]

- Bitstream extraction from HW: [1]
  - Wiretap RAM connection lines on PCB
  - Read flash memory directly
- Bitstream conversion
  - File format is publicly known
  - Correlation between bits and logic is not [4]
  - Given bitstream is compared with a reduced design containing the logic under investigation

Another Approach: Analysing Optical Emissions

- Based on charge carriers emitting photons when being accelerated by electric fields [5]
- Software must solely trigger specific parts of the chip
- Light-sensitive camera setup is needed
- Suitable e.g. for analysing MCUs or SoCs and evaluating the functionality of certain parts

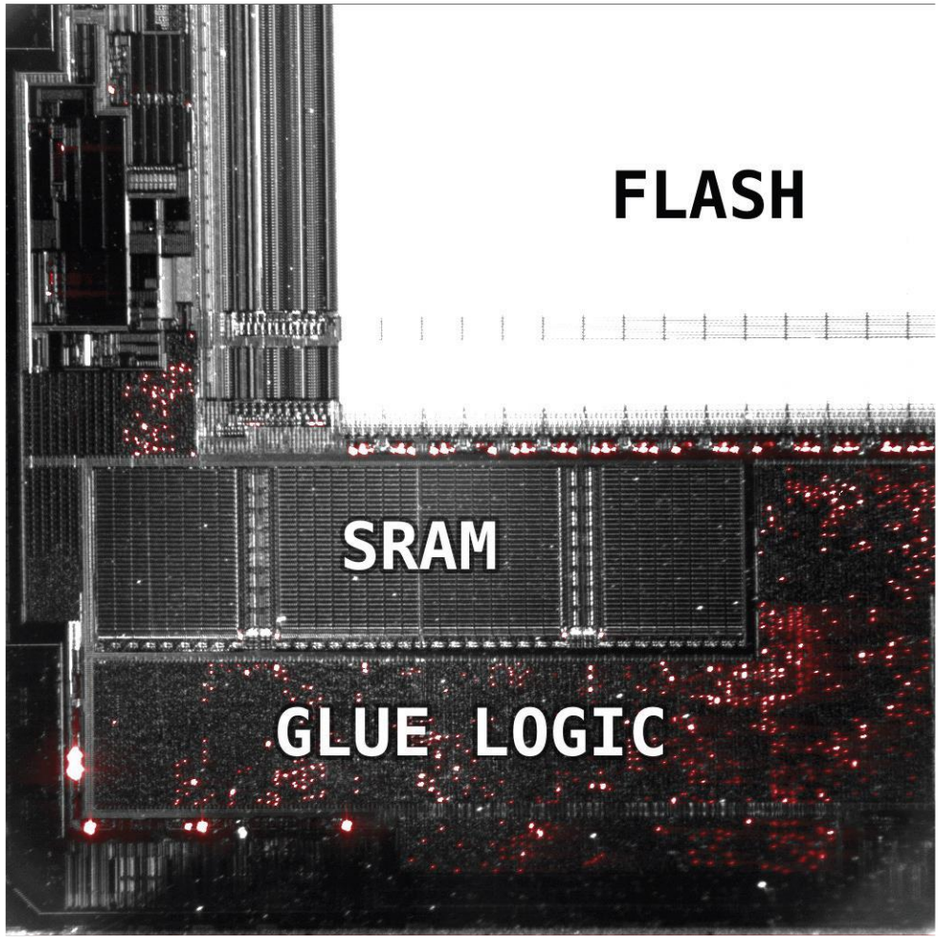


Fig. 9: Internals of an ATmega328p overlayed with optical emissions [5]

Specification Discovery

- Gate-level netlist is taken as input
- Goal is to get an understanding of the functionality of the IC
  - Ideal: high-level RTL or HDL description
- This is still a hard problem
- Graph theory and SAT solvers are essential for this

Circuit Partitioning

- Netlist is partitioned into subblocks to reconstruct original hierarchy
- Needed to handle circuit with millions of gates
- Considers, that chips are divided into submodules during development too
- **Top-Down partitioning** is used
  - Idea: Blocks that belong together have lots of connections in-between
  - e.g. *min-cut* is used

Analysing the Subblocks

Structural Analysis [1]

- Comprehensive library is used to match subcircuits
- Isomorphisms must be kept in mind

Functional Analysis

- Examines logic functions from gate-level netlist
- Assumes, that if two polynomial functions yield equal results for some group of inputs, the corresponding Boolean functions are equal with high probability
- Checks against given library

Postprocessing

- Get an overview of the implemented design
  - How do the identified subblocks work together?
- Several small problems, that all need a specialised approach
  - What is the functionality of the identified modules?
  - How to name modules, nets, connections to ease further evaluation?

Extracting Datapath

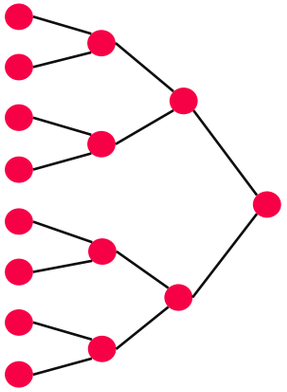


Fig. 8: Tree Structure

- **Word-level Identification** [1]
  - Identifies (shift-) registers, multiplexers, ALUs …
- RAM and register files can be easily identified by their **tree-like structure**
- Coverage of a SoC might be quite high after extracting the datapath
  - ~ 45 – 94 % in evaluated subset
  - But only part of actual functionality of chip

Extracting Finite State Machines (FSMs)

- Important part of control logic
- Graph algorithms can identify **Strongly Connected Components** [1]
- Other features indicating state registers belonging together:
  - Same enable signal
  - Shared gates in feedback path
  - Similar to identical fan-in cones
- Another challenge: identifying control signals

Extracting State Transitions

- Initial states are evaluated
  - Reset behaviour (ASIC)
  - Initial register values (FPGA)
- **Brute-force Method** applies all possible input states [1]
  - Observes state changes by evaluating combinatorial logic in-between the state registers
  - High complexity: $\mathcal{O}(|S| * 2^i)$
- Other approaches with Machine Learning or Structural Analysis

Result

- Now we have a full understanding of the functionality of the IC or a specific part of it
- Afterthoughts:
  - Is the result correct?
  - How does the chip work?
  - How can this knowledge be incorporated into own designs?

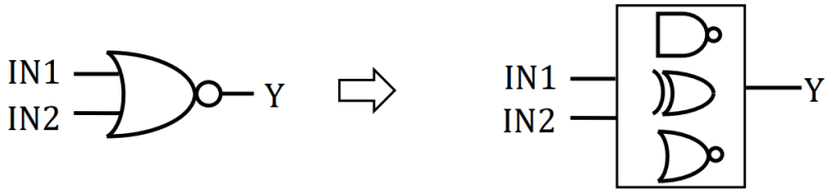
4. Limitations of IC Reverse Engineering

Countermeasures – “Deliberate Obfuscation”

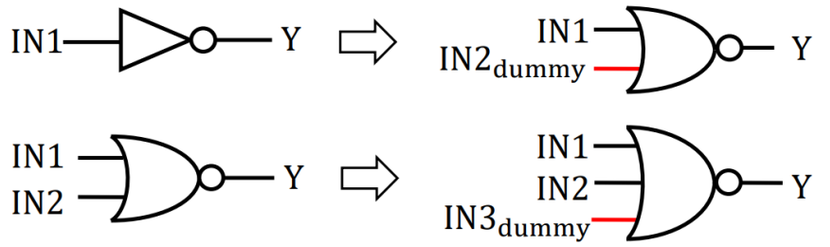
- Logic Locking [2]
  - HW does not work correctly until a secret key is provided
  - XOR logic is added and may obfuscate further by simplifying with actual logic through an additional synthesis step
- Implementing the same HW differently within the same chip
  - Offers no security against functional algorithms
- Full-custom chips
  - Not using standard cells makes it harder for attackers to extract the gate-level netlist

Countermeasures – “Deliberate Obfuscation”

- IC Camouflaging [10]
  - Make cells look like (other) standard cells
  - Dummy contacts are added
  - Gate-level netlist cannot be created from image as functionality of cells cannot be identified
- For FPGAs:
  - Bitstream encryption

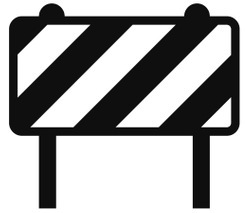


(a) Regular Camouflaging



(b) Proposed Camouflaging

Limitations



- Rapid technological development
- Increasing complexity of modern ICs
- Cost of RE
- Increasing use of deliberate obfuscation and other counter-measures to secure IP [2]

Thank‘s for your attention!

Any questions?

Bibliography

[1] L. Azriel, J. Speith, N. Albartus, *et al:* ”**A survey of algorithmic methods in IC reverse engineering**”, *J Cryptogr Eng* 11, 299–315 (2021). https://doi.org/10.1007/s13389-021-00268-5.

[2] P. Chakraborty, J. Cruz and S. Bhunia: “**SAIL: Machine Learning Guided Structural Analysis Attack on Hardware Obfuscation**”, *2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, Hong Kong, China, 2018, pp. 56-61, DOI: 10.1109/AsianHOST.2018.8607163.

[3] J. Kumagai: “**Chip detectives [reverse engineering]**”, in *IEEE Spectrum*, vol. 37, no. 11, pp. 43-48, Nov. 2000, doi: 10.1109/6.880953.

[4] S. McNeil: “**Solving Today's Design Security Concerns**”, from Xilinx, 2012-06.

[5] D. Nedospasov, A. Schlösser, et. al: "**Functional integrated circuit analysis**“, in *2012 IEEE International Symposium on Hardware-Oriented Security and Trust*, San Francisco, CA, USA, 2012, pp. 102-107, doi: 10.1109/HST.2012.6224328.

[6] R. Torrance, D. James: “**The state-of-the-art in semiconductor reverse engineering**”, in *2011 48th ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2011, p. 333-338.

[7] Wikipedia contributors: “**Reverse engineering --- Wikipedia, The Free Encyclopedia**”, 2023, url: https://en.wikipedia.org/w/index.php?title=Reverse_engineering&oldid=1183051000.

[8] M. Hansen, H. Yalcin, J. Hayes: “**Unveiling the ISCAS-85 benchmarks: a case study in reverse engineering**”, IEEE Des. Test Comput., 16(3), 72–80 (1999).

[9] I. Polian: “**Security aspects of analog and mixed-signal circuits**”, In: 2016 IEEE 21st International Mixed-Signal Testing Workshop, IMSTW 2016. Institute of Electrical and Electronics Engineers Inc. (2016).

[10] B. Shakya, H. Shen, M. Tehranipoor, and D. Forte: “**Covert Gates: Protecting Integrated Circuits with Undetectable Camouflaging**”, TCHES, vol. 2019, no. 3, pp. 86–118, May 2019. doi: 10.13154/tches.v2019.i3.86-118.

[11] T. Weber: **„Reverse Engineering Architecture And Pinout of Custom Asics“,** from SEC Consult Vulnerability Lab, 2019-02, url: https://sec-consult.com/blog/detail/reverse-engineering-architecture-pinout-plc/

[12] V. Raja, K. J. Fernandes: „**Reverse Engineering – An Industrial Perspective**“, in *Springer Series in Advanced Manufacturing*, doi: 10.1007/978-1-84628-856-2_7