Overview of Protections against IC Counterfeiting and Hardware Trojan Horses

Jean-Luc Danger
Outline

- IC Counterfeiting
  - Overview of the threat
  - Detection methods
  - Prevention methods

- Hardware Trojan Horses
  - Types
  - Detection methods
  - Prevention methods

- Conclusions
IC Counterfeiting is a reality

- The IC **Supply chain** (distributors, brokers,…) is an open door to counterfeit components [1] (SIA report)
- All sectors are impacted, including military[2] 5DoD report)

**Economic Harm**
- Reduction of the Original Component Manufacturer (OCM): market share: $169 billion in 2012 [3]

- Damage due to lack of reliability
  - The counterfeit circuit may be defective
  - Negative image of the OCM
IC counterfeiting

IC life cycle:

IC design → Fabrication & test → Supply chain → usage → scrap

Are they trustworthy?

- **Counterfeiting types** [4]
  - The circuit is the **original** one but has been illegally manipulated
  - The circuit is **fake**
Counterfeiting with original circuit

IC life cycle:

- **Recycling**
  - The circuit has been taken from old PCBs and remarked

- **Falsification**
  - The labeling, specification and certification are forged

- **Overproduction**
  - There is no legal contract for fabrication

- **With defects**
  - The component did not pass the tests
Counterfeiting with fake circuit

IC life cycle:

- **IC design**
- Fabrication & test
- Supply chain
- Usage
- Scrap

**Cloning**
- The circuit has been pirated by reverse engineering and redesigned identically

**Hardware Trojan Horse (HTH)**
- The circuit has been tampered at the fab stage and some extra logic called Hardware Trojan Horse has been added to spy or destroy it
How to protect from counterfeiting?

- To work with trusted partners
  - Design House: to make ICs in trusted fabs
  - User: to buy ICs to trusted distributors

- To use detection techniques
  - For existing devices
  - For new devices with dedicated hardware

- To use prevention techniques
  - Only for new devices
3 main types:

- **Physical** analysis
  - Can be destructive
  - Only used to detect recycling and forged circuits
- **Electrical** tests
- **Aging** tests
Detection: Physical analysis

- Imaging
  - Visual inspection
  - XRAY imaging
  - Scanning Acoustic Microscopy (ultrawave)
  - Scanning Electron Microscopy

- Material Analysis
  - XRAY fluorescence spectroscopy
  - IR spectroscopy (IR absorption)
  - THz spectroscopy (absorption in metal)
Example: X-ray Nanotomography[5]
Detection: Electrical tests

- **Integrity tests**
  - Scan chain to detect failures

- **Parametric tests**
  - DC and AC parameters in
  - In various environment
  - To detect abnormal offset

- **Functionnal tests**
  - To detect out of range ICs

- **Burn-in test**
  - Accelerate the **aging** and the failure occurrence
Detection: Aging tests

- **Data analysis**
  - Delay measurement which is very sensitive to aging
  - Machine Learning algorithms used to classify two sets of data:
    - New trusted devices and
    - unknown devices, presumably new
  - This method is impacted by process variation

- **With internal sensors**
  - CDIR (Combating Die and IC recycling)
  - Differential structure
    - Ring oscillator reference vs stressed
  - Usage time measurement
    - A clock counter is stored in NVM or OTP (antifuse)
Prevention: Hardware metering

- **Passive** Hardware metering
  - Every circuit has its own ID, either in
  - Non Volatile memory: can be read or tampered
  - Physically Unclonable Function (PUF), which cannot be reverse engineered
  - An authentication protocol is build with the ID

- **Active** Hardware metering
  - The circuit is initially locked. It is unlocked only if the circuit is authenticated.
Prevention: PUF examples

Arbiter PUF

SRAM PUF

(a) SRAM cell CMOS circuit.
(b) SRAM cell logic circuit.
Prevention: PUF-based authentication

- **Challenge C**
  - User sends Challenge C to IC.
  - Response = PUF(C)

- **Nonce**
  - User sends Nonce to IC.
  - E = ENC_{PUF}(Nonce)

- **Genuine IC**
  - Genuine IC if response in the database.
  - Genuine IC if DEC_{PUF}(E) = Nonce.
Prevention: locking mechanism [6]

- The state graph is locked at power-up, and unlocked with the correct sequence
To secure the manufacturing test, hence counterfeiting with defects

The test is done by using asymmetric crypto

The IC owner unlocked the good IC with a master key

**Prevention: Secure Split Test [7]**

- **Foundry tester**
  - IC in \( \text{RND\_mod} = \text{RSA\_pub}(\text{TKEY}) \)

- **IC Owner**
  - \( \text{TKEY} = \text{RSA\_priv (RND\_mod)} \)

- **Test results**
- **Check OK**
The chip manufacturing is split into two steps.

First step:
- Done by any foundry, not necessarily trusted
- In charge of the "front end of line": gates and first metal layers

Second step:
- Done by a trusted foundry
- In charge of finishing the connections "back end of line"

IARPA established a new program in 2011 based on split manufacturing: "Trusted Integrated Chips" [8]
To secure IP (Intellectual Property) block inside an IC

Many ways to insert the mark:

- GDSII
  - Pattern specific to the design
- FPGA
  - unused LUTs in the bitstream
- HDL
  - Unused part of memory, or truth table combinations
- Synthesis

Prevention: Watermarking [10]
Hiding of the cell layout to prevent reverse engineering by optical inspection

- a: NAND, b: NOR
- c: camouflaged NAND, d: camouflaged NOR
## IC Counterfeiting protection efficiency

<table>
<thead>
<tr>
<th></th>
<th>Recycled</th>
<th>Forged</th>
<th>Over produced</th>
<th>Defective</th>
<th>Cloned</th>
<th>HTH</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Detection</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Physical tests</td>
<td>**</td>
<td>**</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Electrical tests</td>
<td>**</td>
<td>**</td>
<td></td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>Aging tests</td>
<td></td>
<td>*</td>
<td></td>
<td></td>
<td>*</td>
<td></td>
</tr>
<tr>
<td><strong>Prevention</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW metering</td>
<td></td>
<td></td>
<td>*</td>
<td></td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>Secure split test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Split manufacturing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>Watermarking</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Camouflaging</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
</tr>
</tbody>
</table>
Outline

- IC Counterfeiting
  - Overview of the threat
  - Detection methods
  - Prevention methods

- Hardware Trojan Horses
  - Types
  - Detection methods
  - Prevention methods

- Conclusions
HTH: A powerful and pernicious threat

- **HTH:**
  - Insertion in an IC of Hardware unknown to the designer
  - Goal: spying, disturbing, destroying
  - Can be inserted at all the levels of the IC design chain

- It is not only an economic threat, it is also strategic
  - 2007 DARPA program "Trust in Ics"
  - 2011 IARPA program "Trusted Integrated Chips"[8] exploiting split manufacturing

- But it is also a weapon for the designer:
  - Backdoors
HTH Principle

- Two components:
  - TRIGGER: Reads and decode internal and rare state
  - PAYLOAD: Writes internal data
- HTH acts as a probing station, both passive (trigger) and active (payload), and is stealthy
HTH Payload examples[12]

- **Kill switch**
  - Simple payload, disastrous effect as Denial of Services

- **Deteriorate the performances**
  - Accelerate the aging, add extra delays

- **Create leakages**
  - Create an access to secret data, either by a functionnal channel or a side-channel

- **Assist malwares**
  - Exploits a hidden function. The HTH is called backdoor if the designer is the creator.
HTH Triggering examples

- **Combinatorial**
  - Decoding of rare event from multiple nodes
    - trigger = f(nodes)
  - Use significant number of gates

- **Sequential**
  - Decoding of a rare event from sequential variables
    - Trigger = f(nodes, time)
    - Less nodes but a few flip-flops

- **Analog**
  - Use internal sensors and external parameters
    - Example: Trigger temperature > threshold
  - Need few gates
HTH Taxonomy [13]

- **Insertion** stage
  - Specification, design, fabrication, test, assembly

- **Abstraction** level
  - RTL, gate, layout, physical,

- **Trigger** type
  - Combinatorial, sequential, analog

- **Payload** type
  - New behavior, less performance, leakages, DoS

- **Physical** characteristics
  - Size, distribution, parametric, functionnal, same layout
HTH protection overview [14]

HT Protection

Post Production Detection
- Destructive
  - Optical
- Non-Destructive
  - Run-Time
  - Test-Time

Prevention
- Supportive Design
- Secure Design
- Trusted Production

Side Channel Analysis
Logic Testing
HTH detection by optical method[15]

- Needs a GDSII golden model

- Comparison between an original GDSII and a trojaned IC with a ×150 lens confocal microscope

Trojan size = 1 AND gate

Trojan size = 128 AND gate

AES

AES
HTH detection by optical method

- Cross correlation between the original AES layout and an affected AES layout

<table>
<thead>
<tr>
<th>Core utilization rate</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>50%</td>
<td>0.9991</td>
<td>0.9972</td>
<td>0.9981</td>
<td>0.9950</td>
<td>0.9933</td>
<td>0.9918</td>
<td>0.9815</td>
<td>0.9668</td>
</tr>
<tr>
<td>60%</td>
<td>0.9987</td>
<td>0.9968</td>
<td>0.9959</td>
<td>0.9955</td>
<td>0.9944</td>
<td>0.9893</td>
<td>0.9788</td>
<td>0.9670</td>
</tr>
<tr>
<td>70%</td>
<td>0.9989</td>
<td>0.9981</td>
<td>0.9918</td>
<td>0.9941</td>
<td>0.9881</td>
<td>0.9850</td>
<td>0.9594</td>
<td>0.9067</td>
</tr>
<tr>
<td>80%</td>
<td>0.9999</td>
<td>0.9965</td>
<td>0.9898</td>
<td>0.9957</td>
<td>0.9780</td>
<td>0.9711</td>
<td>0.8970</td>
<td>0.8509</td>
</tr>
<tr>
<td>90%</td>
<td>0.9988</td>
<td>0.9990</td>
<td>0.9983</td>
<td>0.9962</td>
<td>0.9832</td>
<td>0.9572</td>
<td>0.8858</td>
<td>0.4010</td>
</tr>
<tr>
<td>95%</td>
<td>0.9997</td>
<td>0.9984</td>
<td>0.9980</td>
<td>0.9889</td>
<td>0.9589</td>
<td>0.9115</td>
<td>0.8824</td>
<td>0.8202</td>
</tr>
<tr>
<td>99%</td>
<td>0.9917</td>
<td>0.938</td>
<td>0.9714</td>
<td>0.9527</td>
<td>0.3798</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
</tbody>
</table>

In black: ECO routing

- Trojan almost impossible to insert without changing the layout, if occupancy rate > 80%
HTH detection at test time

- **Logic Testing**
  - To search the triggering of the Trojan
  - Need exhaustive search of a rare event => impossible
  - Rather use statistical approaches as MERO[15]
  - Or add HW to avoid rare event

- **Side Channel**
  - To detect the resources of the Trojan
  - By measuring:
    - the Current (IDDQ, IDDT)
    - the EM field
    - the propagation delays
  - Very sensitive to noise and process variance
HTH detection with side-channel[17]

- Needs a golden model of the "activity"
- Measurement of local EM field with RF probes
- Impact of noise => Probability of detection
HTH detection with side-channel

- Impact of process variance
  - HTH of different sizes: HTH greater than 1% can be detected with a false negative rate of 0.017%.
HTH detection at run time

- Techniques to check the integrity in real time
- Can take advantage of SEU and attack detection techniques:
  - Error correction codes
  - Control Flow Integrity (processors)
  - Hardware Assertions checkers
  - Real time security monitor
HTH detection by flow integrity check[18]

- The processor control flow can be tampered by HTH and/or malwares
- Prevention can check the integrity of basic blocks and unexpected jumps
  - Example: Use golden tables of basic blocks CRC and jump tables
Example: The HTH outputs a secret key with on the UART channel by doubling the Baudrate

Property to check by Hardware:
- The serial bits have to be stable during a fixed period.
- If the baudrate changes, the assertion fails
HTH prevention

- **Split manufacturing**
  - Use a "root of trust" with two steps: Front-end of line, and Back end of line

- **To use the layout-filler**
  - No more places to insert HTH on GDSII

- **To avoid rare events** during test time

- **Obfuscation**
  - To obfuscate the state transitions by keys
    - Active **Hardware metering**
  - To obfuscate by error correcting codes (ECC)
    - Mask the signals with random variables
Encoding the circuit [20]

- **Principle:**
  - The HTH has two parts:
    - probing (trigger) and fault injection (payload)
  - The registers are the most easy cells to detect, thus the most easy to probe for Trojan insertion
  - The sequential variables in registers are encoded by Linear Complementary Dual Codes (LCD)
  - The Dual code allows the designer to use random variables to mask the real computation

- **Protection also effective against:**
  - Probing attacks
  - Fault attacks
  - Side-Channel Attacks
Encoding the circuit: Architecture
Encoding the circuit: Methods

- **G** encodes **k** bits
- **H** is the dual of **G**
  - \((GH^T = 0)\)
- **Encoding:**
  - \(Z = xG \oplus yH\)
  - \(X = \text{information}\)
  - \(Y = \text{random variable}\)
- **Decoding\(^*\)**
  - \(J = G^T (GG^T)^{-1}\)
  - \(K = H^T (HH^T)^{-1}\)

\(^*\) Moore-Penrose pseudo-inverse
The code \([n,k,d]\) has a proven security of \(d\):
- The HTH \textit{trigger} is inefficient with less than \(d\) probes
- The HTH \textit{payload} is inefficient if it modifies less than \(d\) nets

\begin{itemize}
  \item \textbf{Complexity}
  \begin{itemize}
    \item Choose low density codes to encode and decode
  \end{itemize}
\end{itemize}
Encoding complexity

Table 8.6 – Synthesis results of encoded circuit method, and security parameters for the SIMON co-processor.

<table>
<thead>
<tr>
<th>IC (Code)</th>
<th>Gates</th>
<th>Area ($\mu$m$^2$)</th>
<th>n</th>
<th>k</th>
<th>$d_{\text{Trigger}}$</th>
<th>$d_{\text{Payload}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original ([109,109,1])</td>
<td>300</td>
<td>1919</td>
<td>109</td>
<td>109</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Encoded ([110,109,2,1])</td>
<td>560</td>
<td>3567</td>
<td>110</td>
<td>109</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Encoded ([140,109,10,6])</td>
<td>3107</td>
<td>20239</td>
<td>140</td>
<td>109</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>Encoded ([123,109,5,3])</td>
<td>2348</td>
<td>15249</td>
<td>123</td>
<td>109</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>
Conclusions

- Methods for counterfeiting and HTH insertions are sophisticated and increasing.
- Many protections:
  - But need resources:
    - Tools and methods for detection
    - Extra Silicon and methods for prevention
    - Split foundries
  - The optimal solution is still a challenge
    - Combination of techniques
    - With reduced complexity to get higher detection or avoidance rate
  - But very few inputs from the industrials
Key References