Hardware Trojans and Other Threats against Embedded Systems

Janet Lackey under CC license

ASIA-CCS

Abu Dhabi, April 3, 2017

Christof Paar

Ruhr Universität Bochum

Acknowledgement

- Georg Becker

- Pawel Swierczynski

- Marc Fyrbiak

Agenda

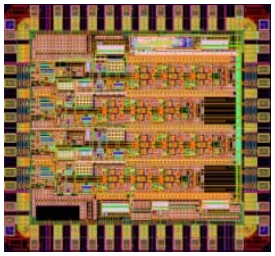
- Introduction to Hardware Trojans
- Sub-Transistor ASIC Trojans
- FPGA Trojan
- Key extraction attack
- Auxiliary Stuff

Agenda

- **Introduction to Hardware Trojans**
- Sub-Transistor ASIC Trojans
- FPGA Trojan
- Key extraction attack
- Auxiliary Stuff

Hardware Trojans

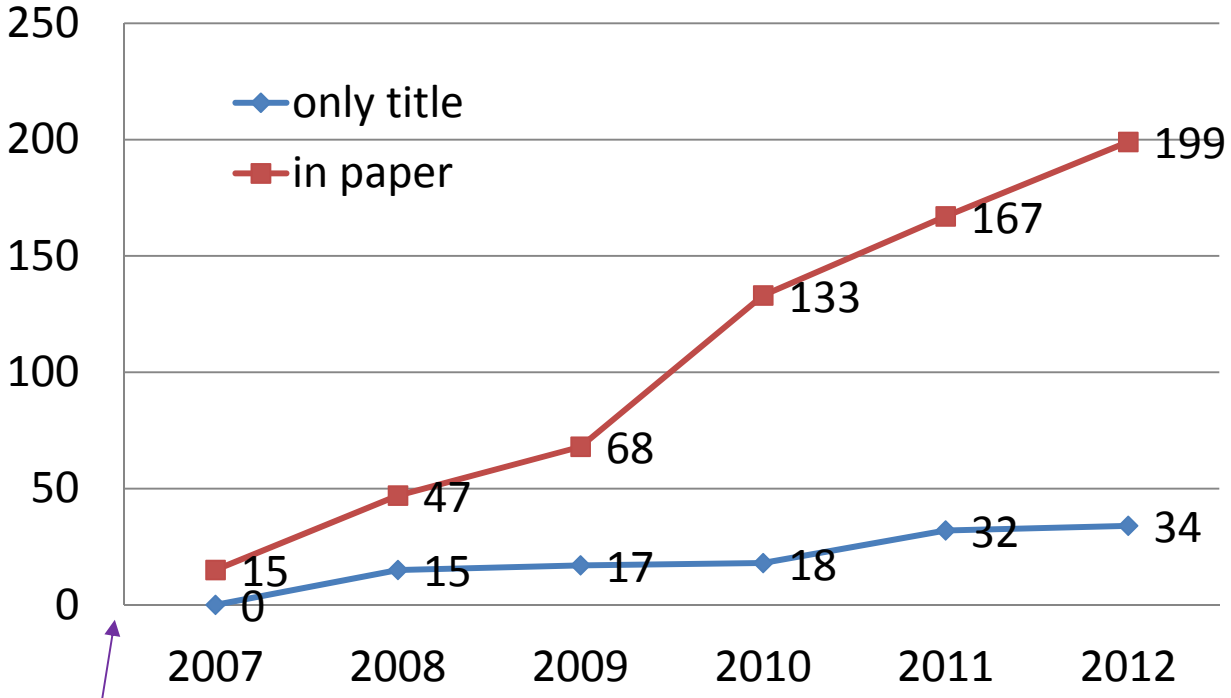
Malicious change or addition to an IC that adds or remove functionality, or reduces reliability



Many rather unpleasant “applications”

Hardware Trojans & the Scientific Community

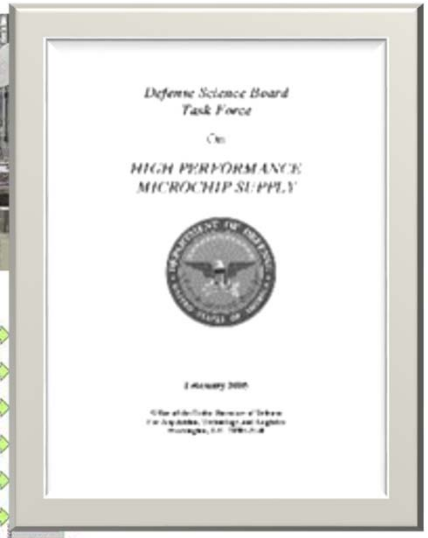
Publications w/ „Hardware Trojans“ or „malicious Hardware“
(Google Scholar, Aug 2013)

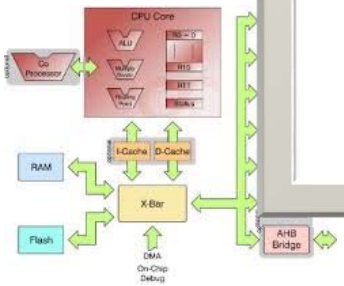


Trojan Injection & Adversaries Scenarios

RUB

DoD scenario 2005





- **Manufacturing**
  Malicious factory, esp. off-shore (foreign Government)
- **Design Manipulation**
  - 3rd party IP-cores
  - malicious employee

not-so-unlikely 2013

- **During shipment**
  cf. NSA's *interdiction*
- **Built-in**
  backdoors etc.

Where are we with “real” HW Trojans?

RUB

- No true hardware Trojan observed in the wild

- All examples from academia

- Vast majority of publications focus on detection

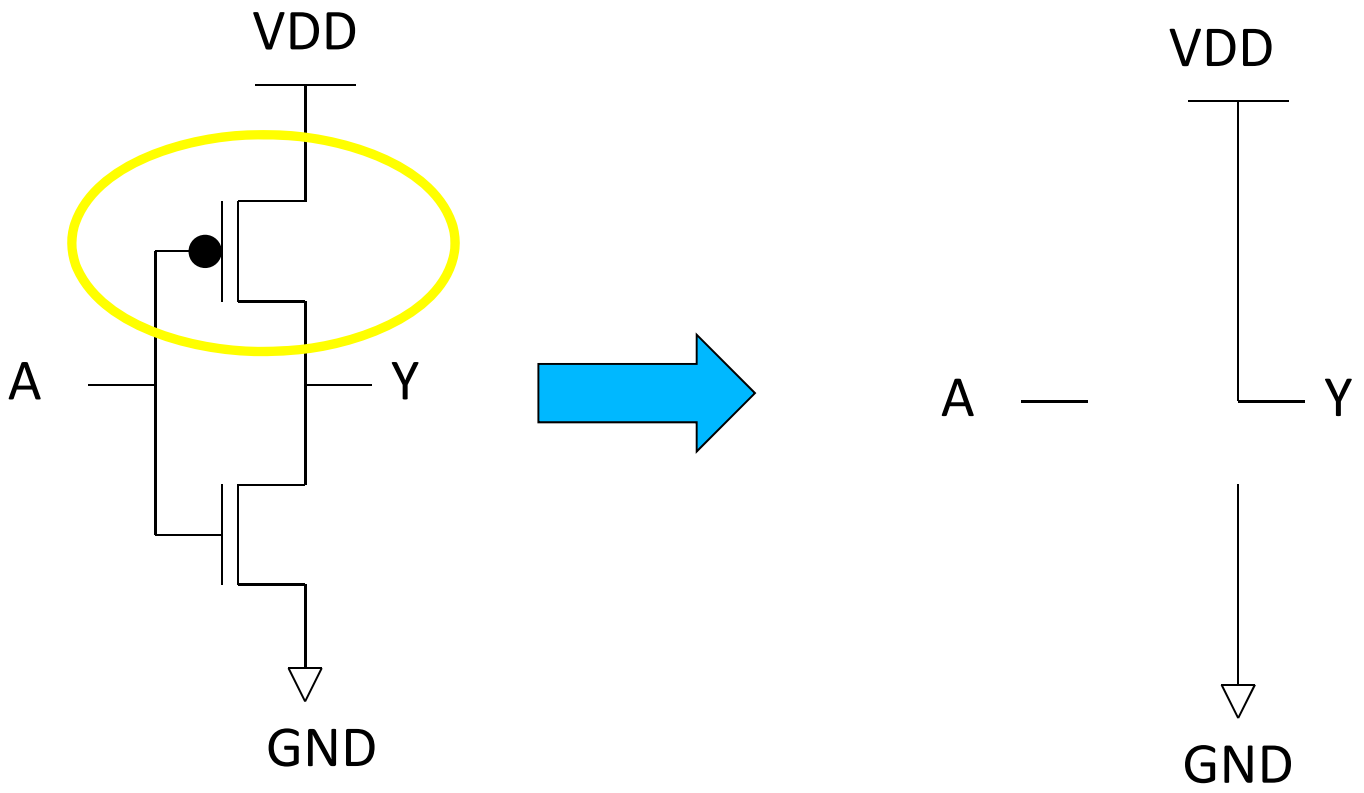
Our Thoughts ca. 2012

1. *Designing* Trojan could be fun too
2. Especially those that go *undetected*

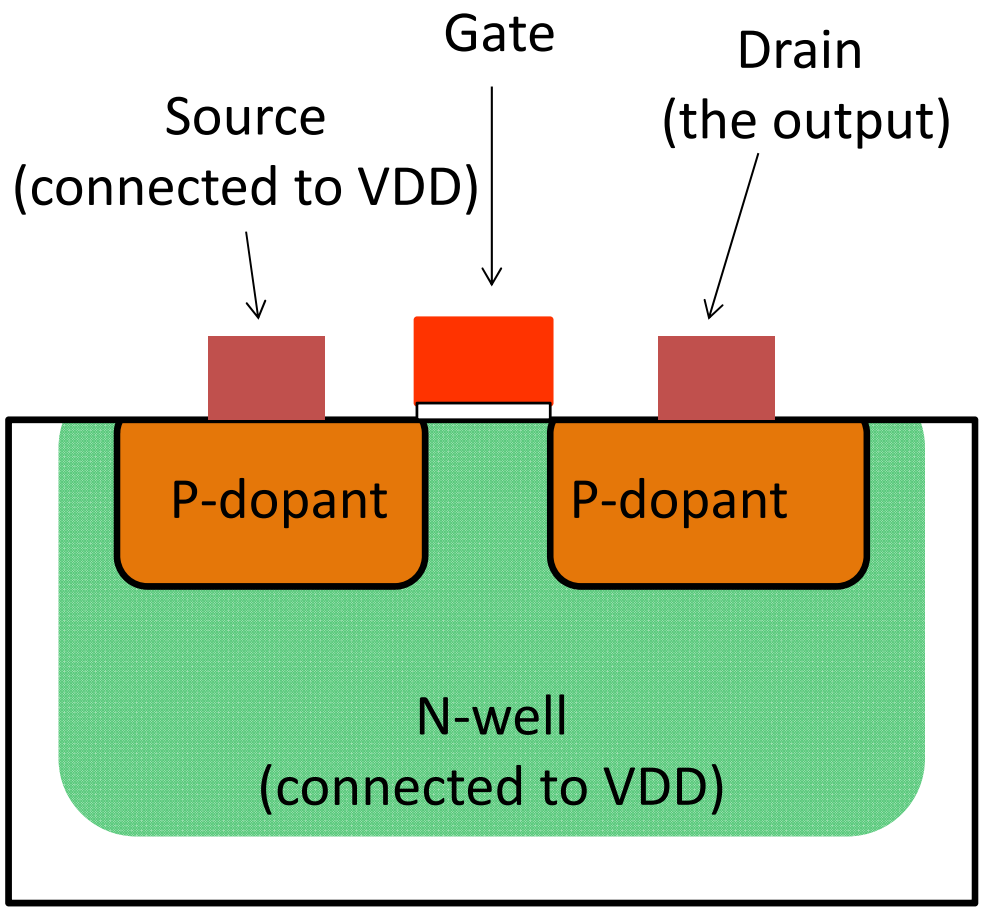
Simple Example: Inverter Trojan

Let’s modify an inverter so that it always outputs “1” (VDD) **without visible changes**.

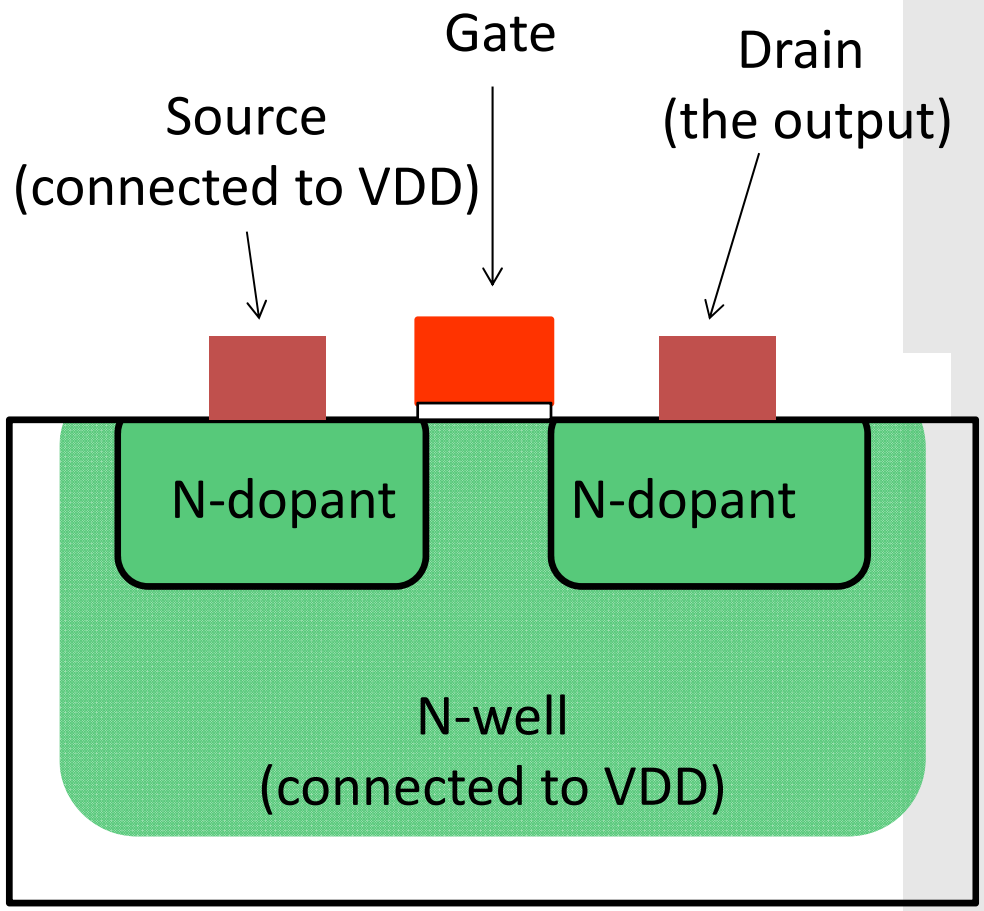
A	Y
0	1
1	0



PMOS Transistor Trojan



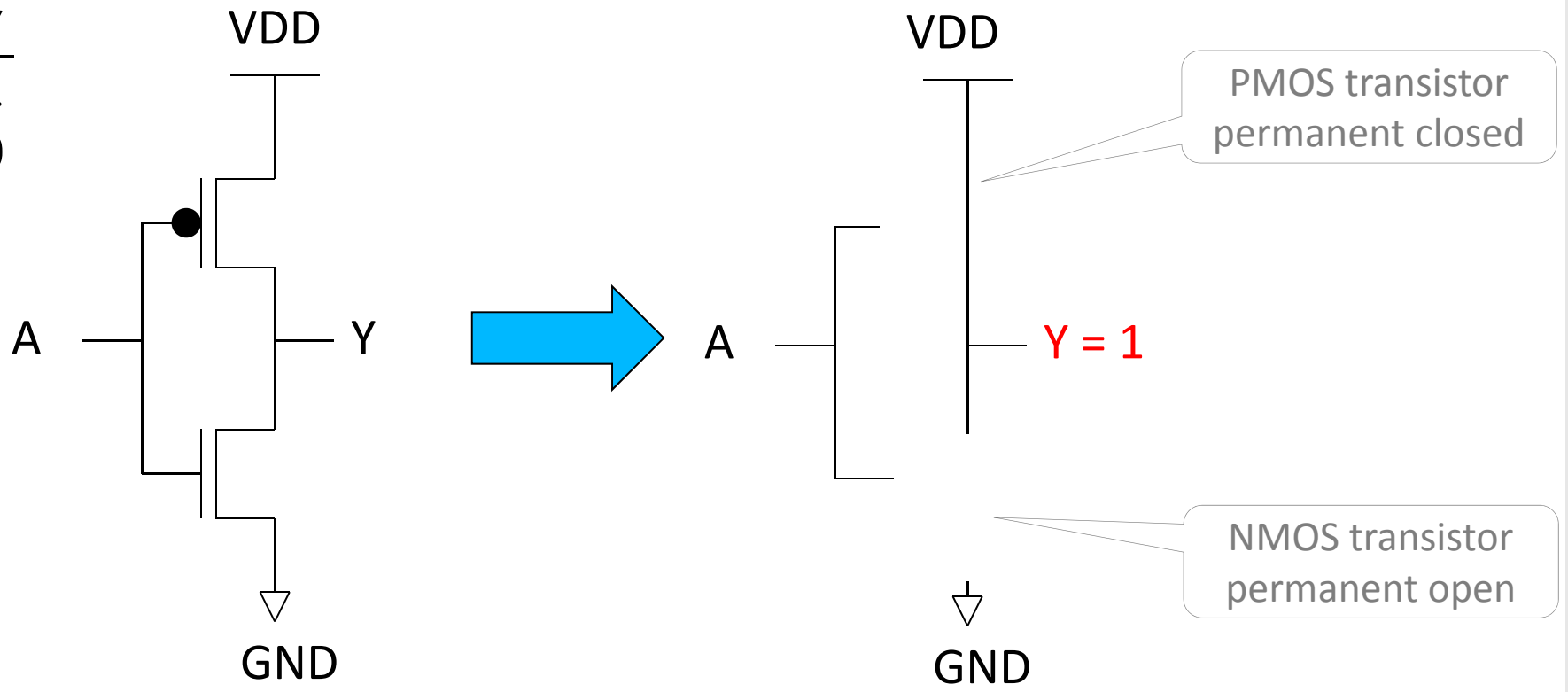
Unmodified PMOS transistor



Trojan trans. w/ constant VDD output

“Always One” Trojan Inverter

A	Y
0	1
1	0



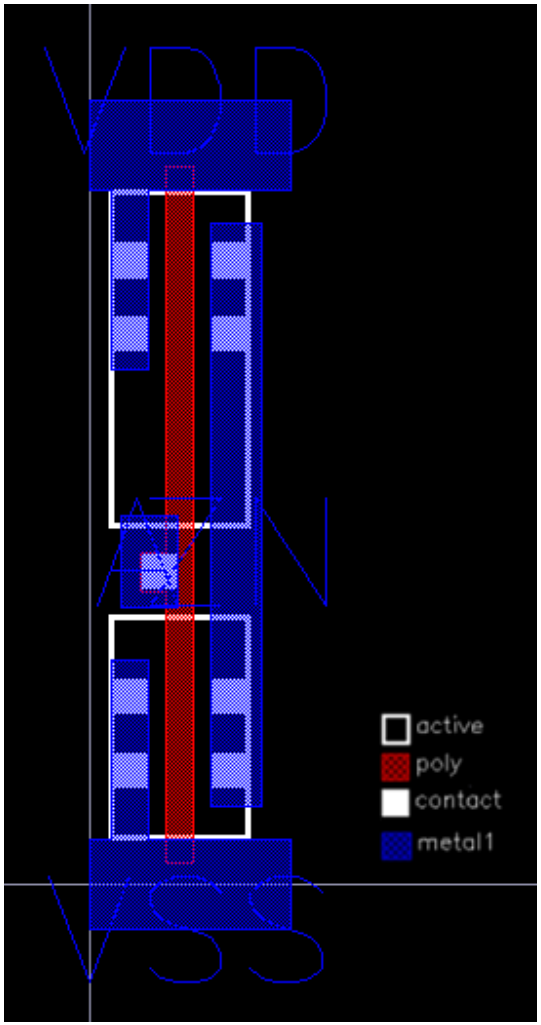
Q1: Can the manipulation be detected?

Q2: How to build a useful Trojan from here?

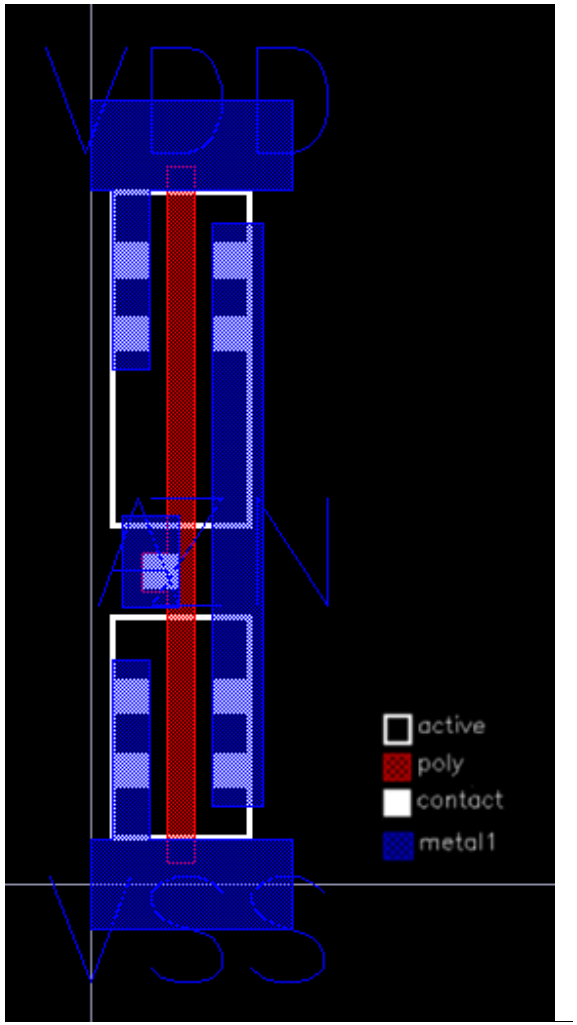
Detection: layout view of Trojan inverter

Which one has the Trojan?

Original Inverter



“Always One” Trojan



Unchanged:

- All metal layers
- Polysilicon layer
- Active area
- Wells

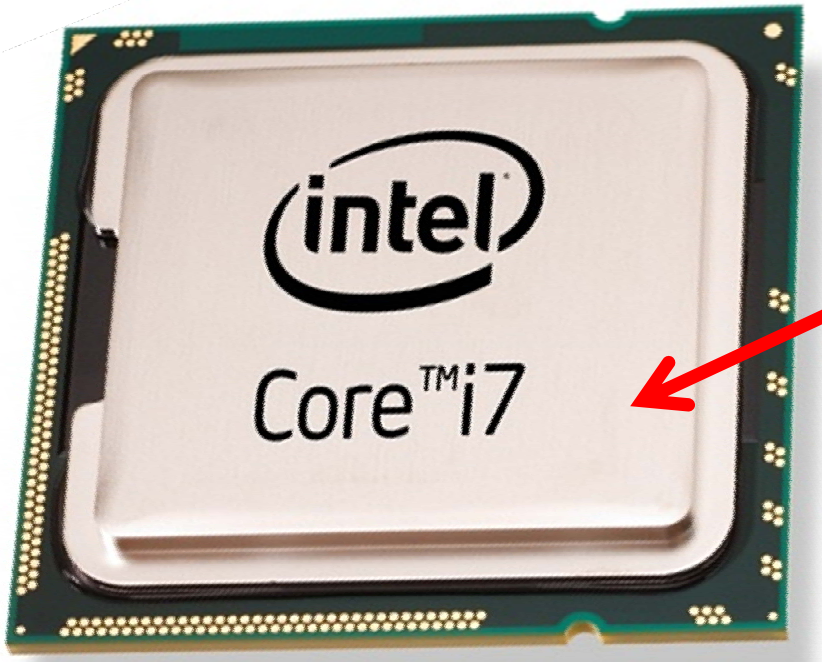
$\Rightarrow$ Dopant changes (very ?) difficult to detect using optical inspection!

“Small” remaining question

- Unfortunately, circuits will not function correctly with this simple stuck-at fault …
- … functional testing (after manufacturing) will detect fault right away

Q2: Can we build a **meaningful** Trojan using dopant modifications that passes functional testing?

Intel's True Random Number Generator

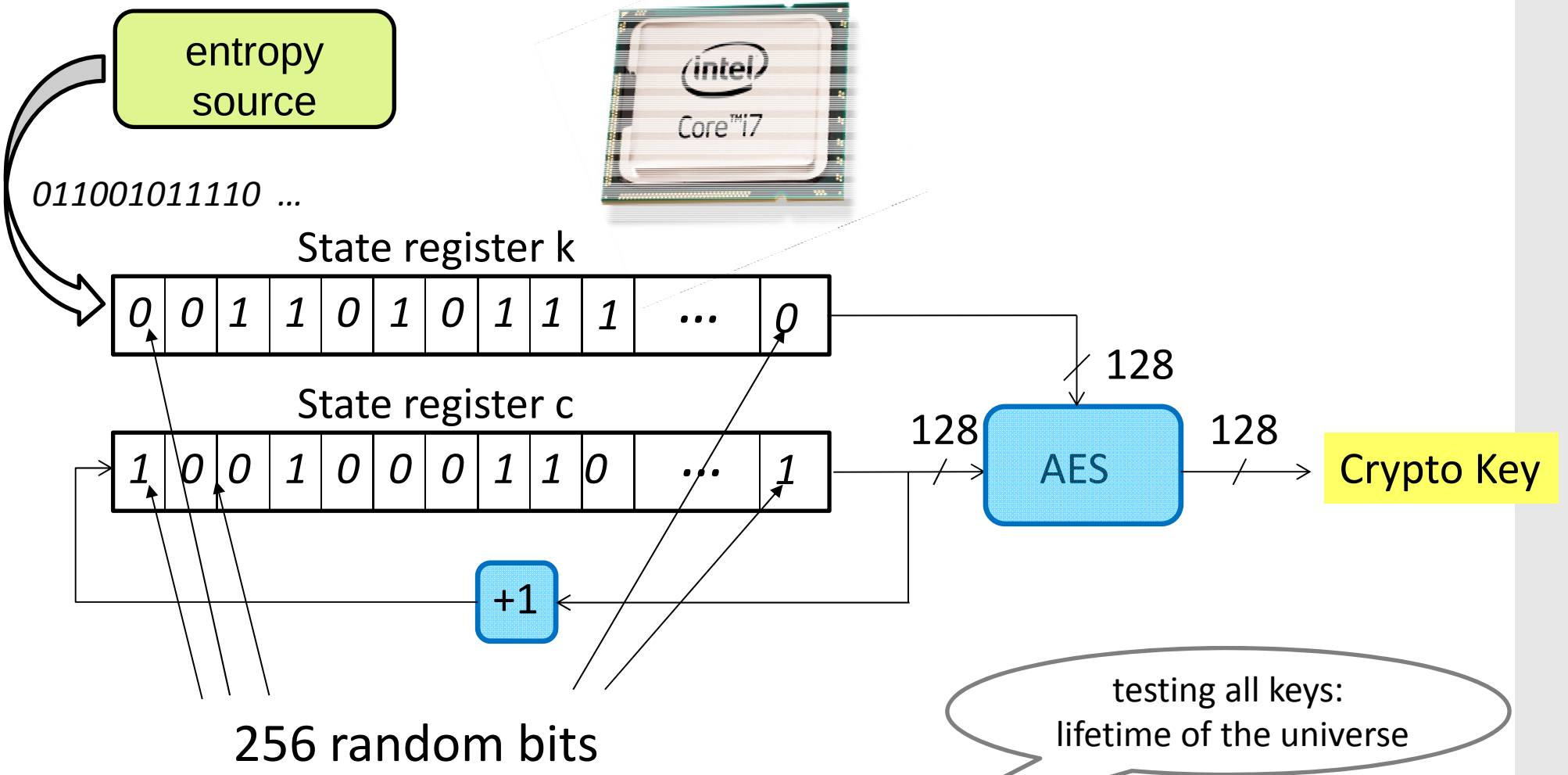


Dopant Trojan

… random numbers generate cryptographic keys for

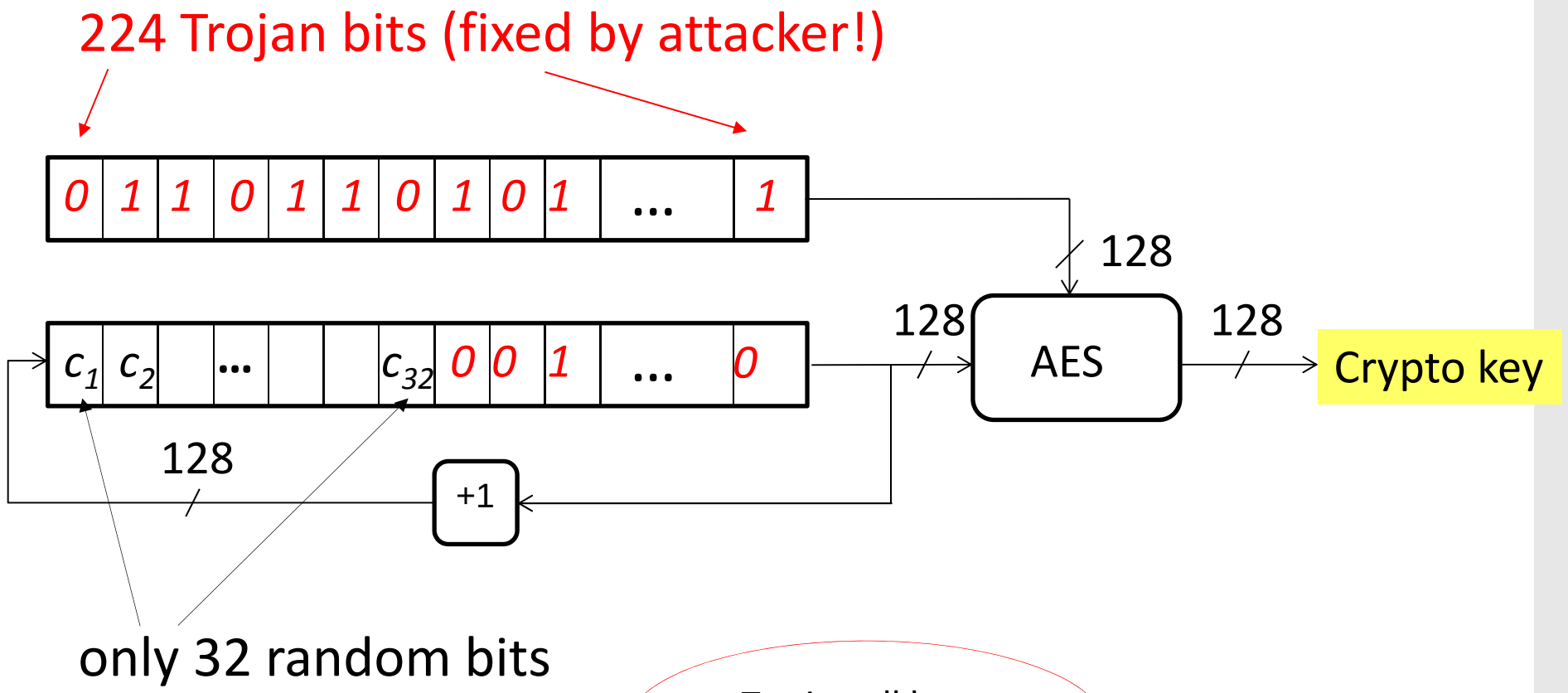
- secure web browsing
- email encryption
- document certification
- …

Inside the Random Number Generator



- 1,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000 possible crypto keys

Trojan Random Number Generator

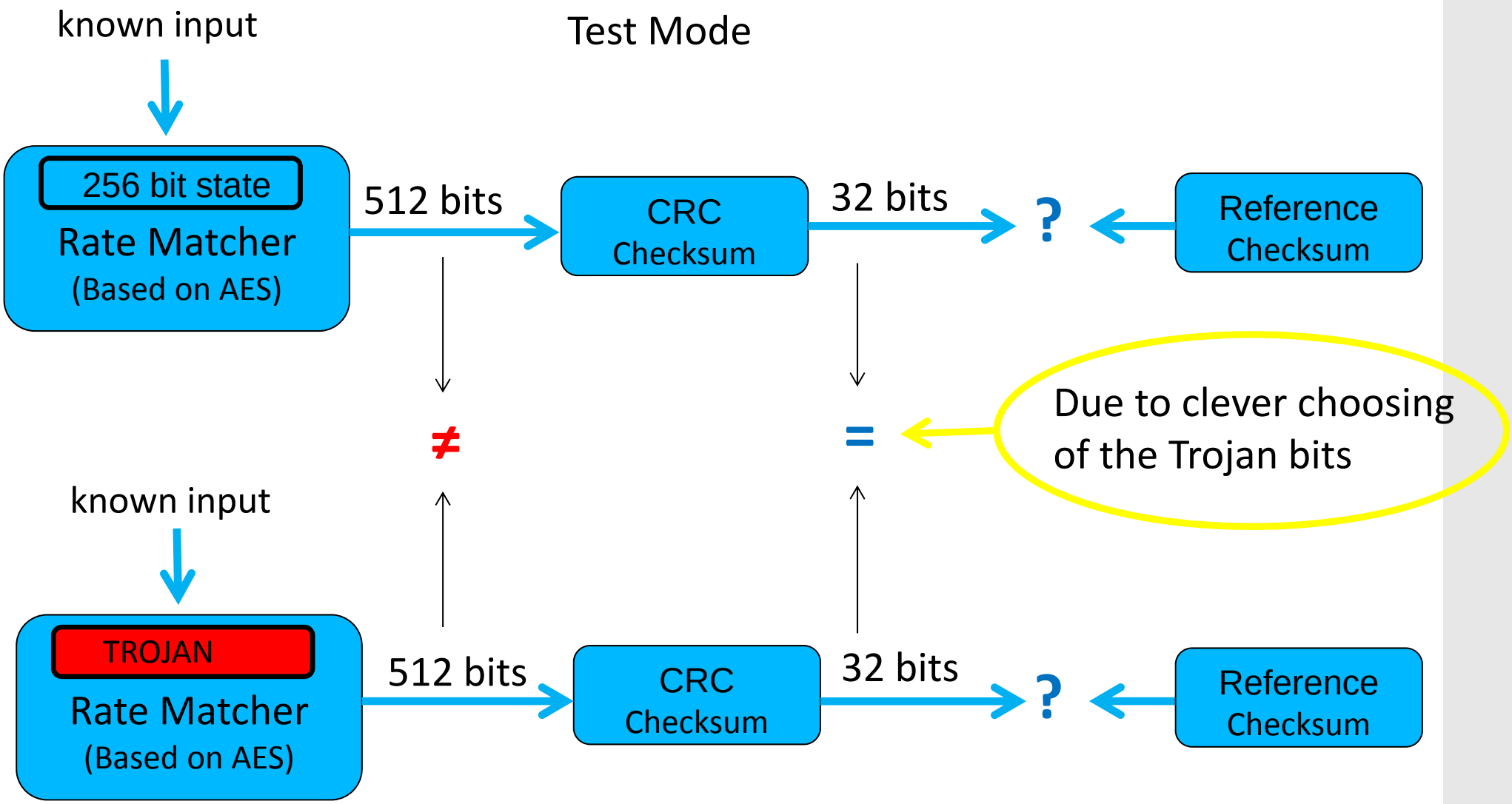


Testing all keys:
few seconds

- 1,000,000,000,000,000,000,000,000,000,000,000,000,000,000 possible crypto keys

... but circuit would still be tested as “faulty” during manufacturing…

Detection prevention through built-in self test



Conclusion

- Meaningful hardware Trojans are possible without extra logic
- Many detection techniques don't guarantee a Trojan free design!
- Built-in self tests can be dangerous
- More details:
  Becker, Regazzoni, P, Burleson, *Stealthy Dopant-Level Hardware Trojans.* CHES 2013

… but the scientific community functions as it is supposed to do:

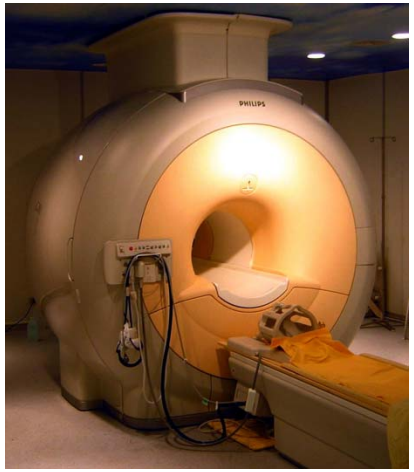
- Trojan detection is possible w/ scanning electron microscope
  Sugawara et al., *Reversing Stealthy Dopant-Level Circuits. CHES 2014*

Agenda

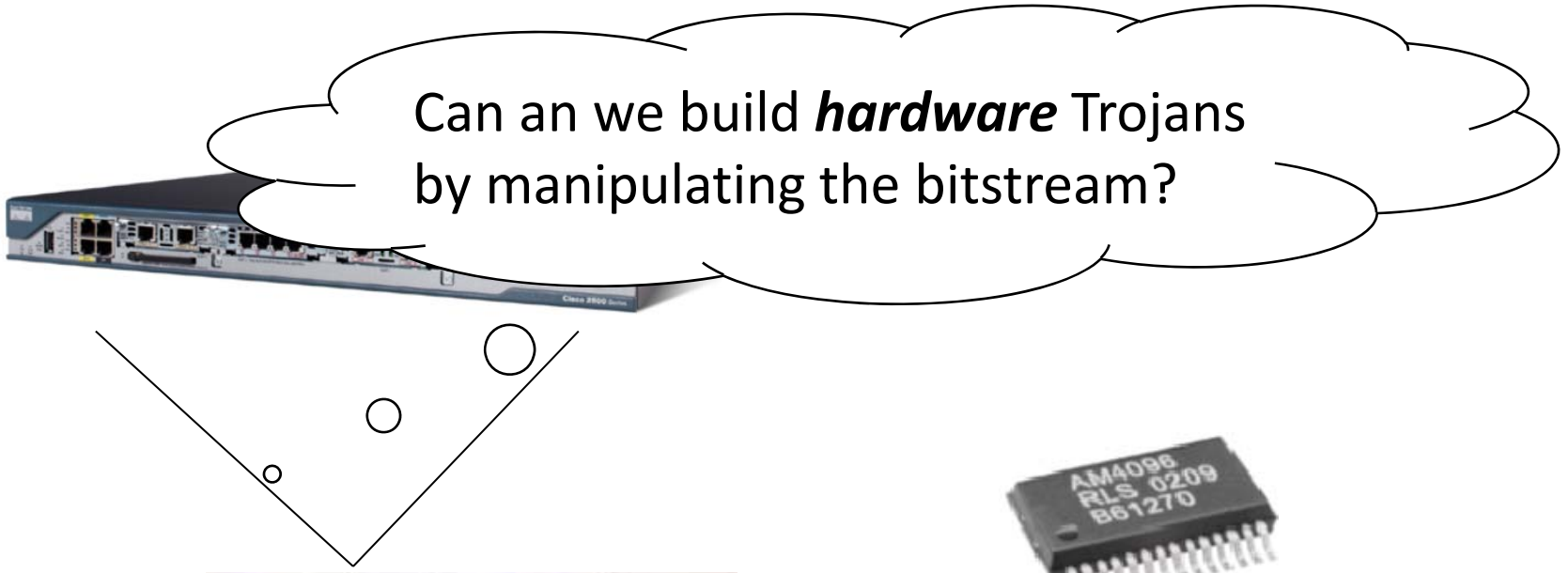
- Introduction to Hardware Trojans
- Sub-Transistor ASIC Trojans
- **FPGA Trojan**
- Key extraction attack
- Auxiliary Stuff

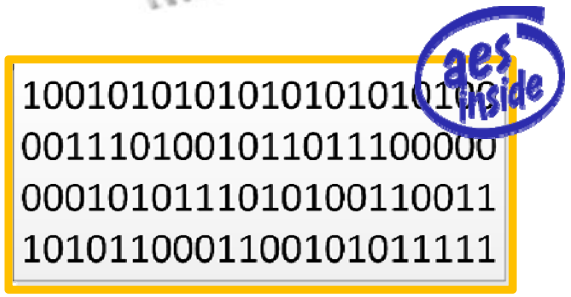
FPGAs = Reconfigurable Hardware
… are widely used

world market:
≈ 5b devices



Configuration during power-up

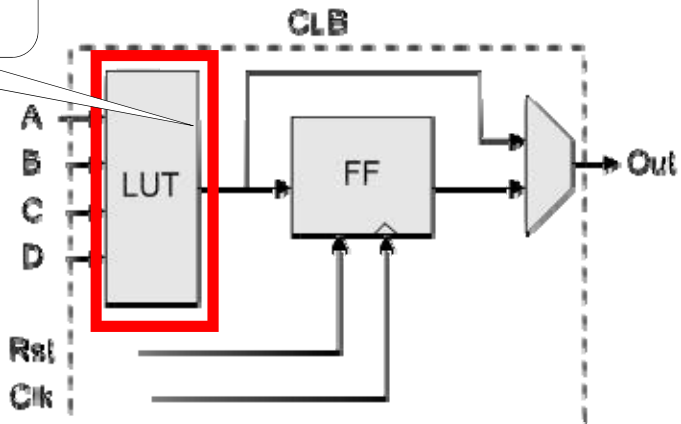




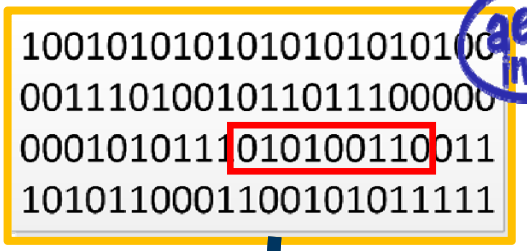
Configuration file
“bitstream”

Principle of FPGA-based Trojans

small look-up tables realize logic



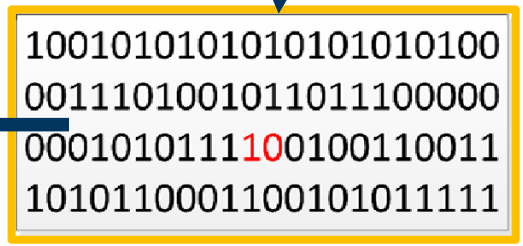
aes inside



aes inside

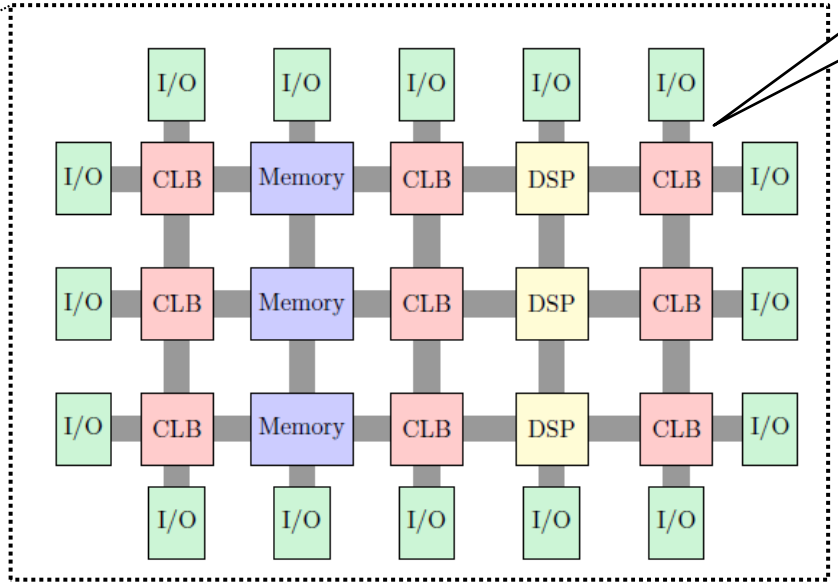
Manipulate Bits

configure



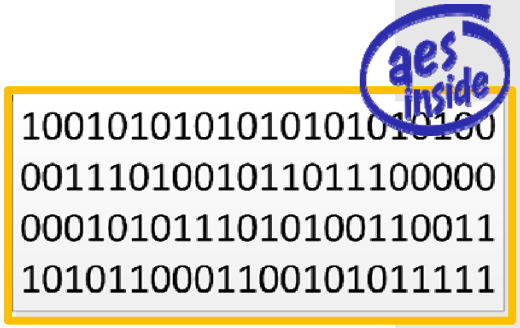
Source Graphics: SimpleIcon, Xilinx

The Mechanics of FPGAs



FPGA fabric

10^3 … 10^6 logic cells

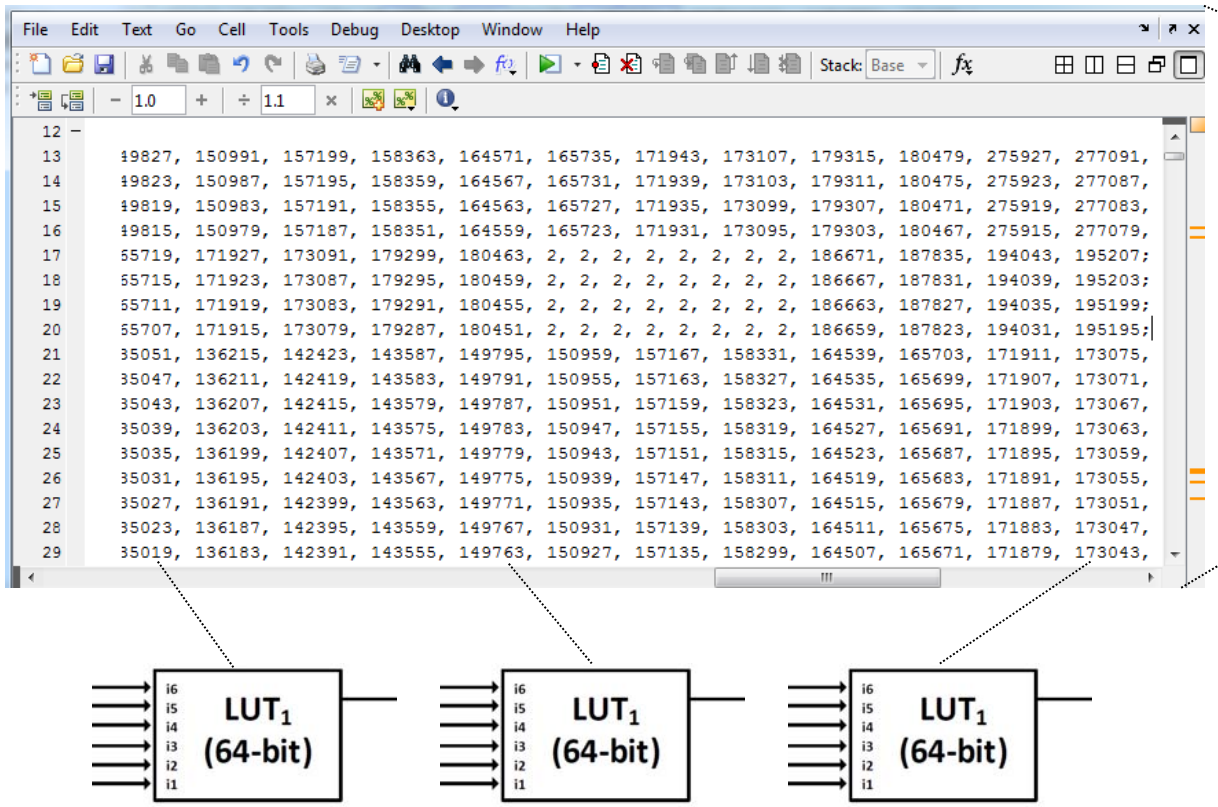


bitstream is complex and proprietary

Two challenges

1. find AES in unknown design
2. meaningful manipulation

Finding AES:
Luckily, crypto has very specific components



- S-boxes are realized as 6x1 look-up tables (LUTs)
- LUT locations can be found in bitstream
- S-box contents is very specific (luckily)

AES detection in practice

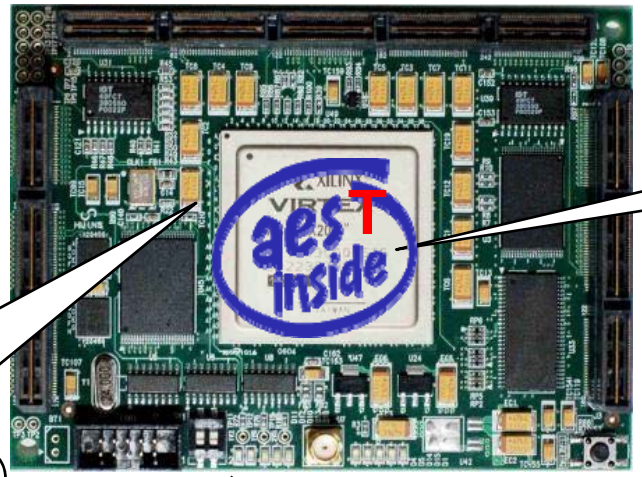
RUB

8 different real-world AES implementations

Impl.	Architecture	AES	LUTs with S-box logic	S-boxes in memory	Detection
#1	Round-based	128	**(16+4)**·32 = 640	no	100 %
#2	$\frac{1}{4}$ Round	128	0	yes	100 %
#3	$\frac{1}{4}$ Round	192	0	yes	100 %
#4	$\frac{1}{4}$ Round	256	0	yes	100 %
#5	Round-based	128	**(0+4)** · 32 = 128	yes	100 %
#6	Round-based	128	0	yes	100 %
#7	Round-based	128	0	yes	100 %
#8	Round-based	128	**(16+4)**·32 = 640	no	100 %

TABLE IV: Overview of evaluated AES implementations

Algorithm substitution attack and its implications



1. Inject **weak S-boxes** in bitstream

2. **Trojan AES** is configured

cute work … but not interoperable with regular AES

PT *CT* = AEST (*k*, *PT*)

"Useful" attacks are still possible!

1. **Storage encryption** (Cloud, harddisk etc.)
   - Attacker can recover plaintext without access to *k*

2. **Temporary access to device**
   - switch S-box and recover *k* from *CT*
   - configure orginal S-box

Conclusion

RUB

- New attack vector against FPGAs!
- Reconfigurability allows “hardware” Trojans designed in the lab
- Bitstream protection is crucial!
  (but not easy, cf. our work at CCS 2011 & FPGA 2013)
- Details at:
  Swierczynski, Fyrbiak, Koppe, P, *FPGA Trojans through Detecting and Weakening of Cryptographic Primitives*. IEEE TCAD 2015.

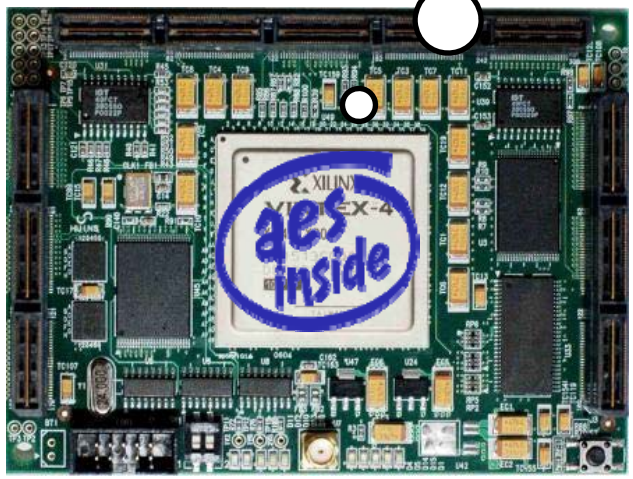
Agenda

- Introduction to Hardware Trojans
- Sub-Transistor ASIC Trojans
- FPGA Trojan
- **Key extraction attack**
- Auxiliary Stuff

What else can we do with bitstreams?

So, bitstream manipulation
allows Trojan insertion ...

Hmm, can we also
extract keys through bitstreams



Set-Up

RUB

non-classical set-up:
Alteration of bitstream

configure

```
10010101010101010100
00111010010111011000000
00010101110101001100111
10101100011001010111111
```

PT

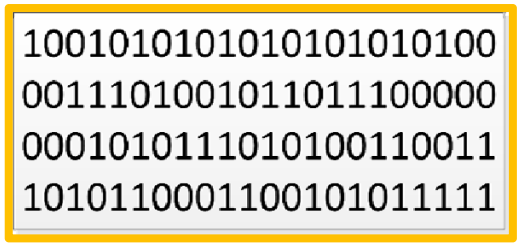
CT = AES (*k*, *PT*)

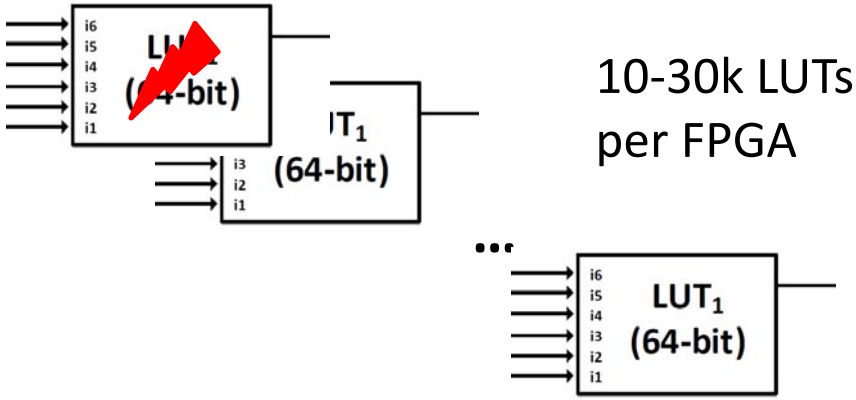
classical known-plaintext set-up

Can Bitstream manipulation of **unknown** design lead to key leakage?

Bitstream Fault Injections (BiFI)

configure





10-30k LUTs per FPGA

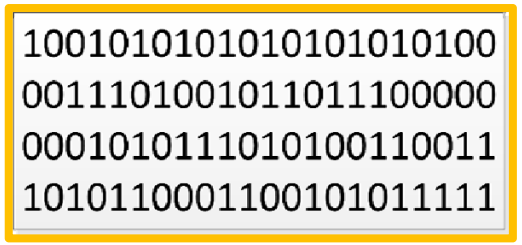
PT *CT* = AES (*k*, *PT*)

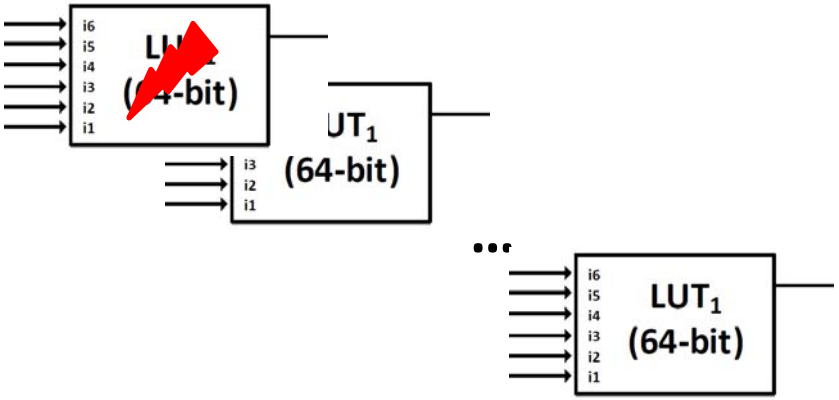
(surprising) attack strategy

1. manipulate 1st LUT table (e.g., all-zero)
2. configure FPGA
3. send PT
4. check: Does CT contain k?
   if not: GOTO 1 and manipulate next LUT

How exactly does the key leak ??

configure





PT *CT* = AES (*k*, *PT*)

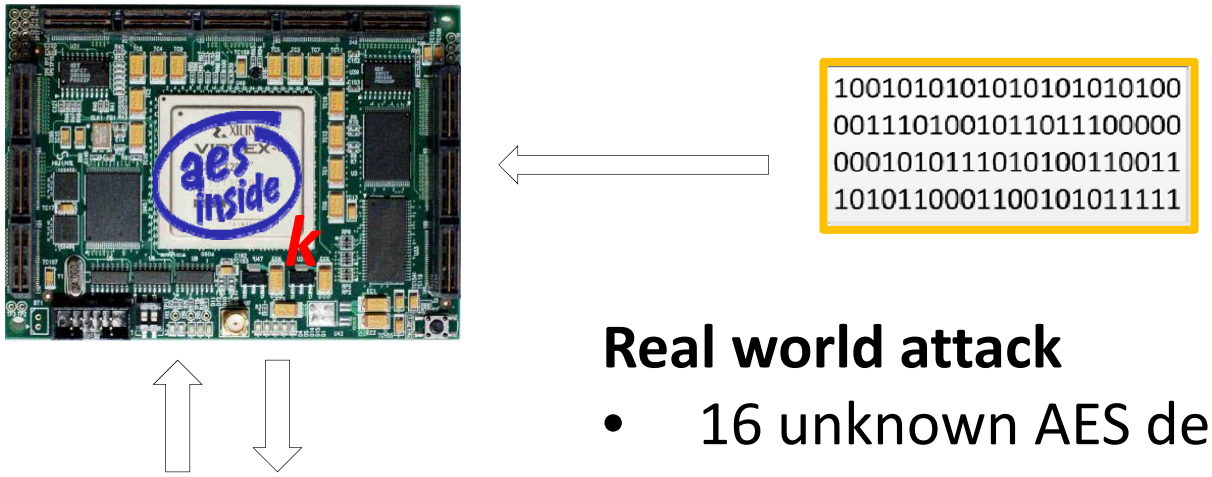
Many leakage hypotheses

- CT = roundkey
- CT = inverted roundkey
- CT = PT xor roundkey
- …

Many LUT manipulations possible

- all-zero
- all-one
- invert
- upper half of LUT all-zero
- …

Results for Bitstream Fault Injections (BiFI)



Real world attack

- 16 unknown AES designs (Internet)
- 16 different manipulation rules
- ≈ 20k LUTs
- 3.3 sec for configuring and checking one alterations

Results

- successful key extraction for **every** design!
- on average ≈ 2000 configurations (≈ 2h)
- works even for encrypted bitstream (w/o MAC)

Conclusion

RUB

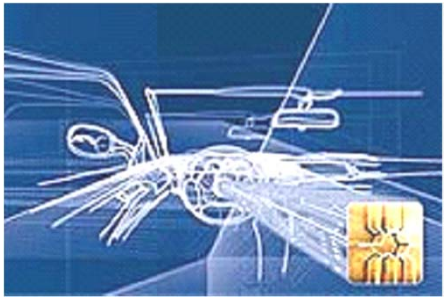
- Bitstream Fault Injections (BiFI) is a new family of fault attacks
- Malleability of bitstream is major weakness for FPGAs!
- Are there more bitstream-based attacks ?
- Details at:
  Swierczynski, Becker, Moradi, P: Bitstream Fault Injections (BiFI) – Automated Fault Attacks against SRAM-based FPGAs. IEEE Transactions on Computers, to appear.

Agenda

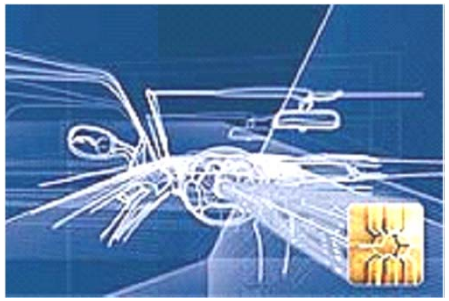
- Introduction to Hardware Trojans
- Sub-Transistor ASIC Trojans
- FPGA Trojan
- Key extraction attack
- **Auxiliary Stuff**

Related Workshops

CHES – Cryptographic Hardware & Embedded Systems
25.-28. September 2017, Taiwan



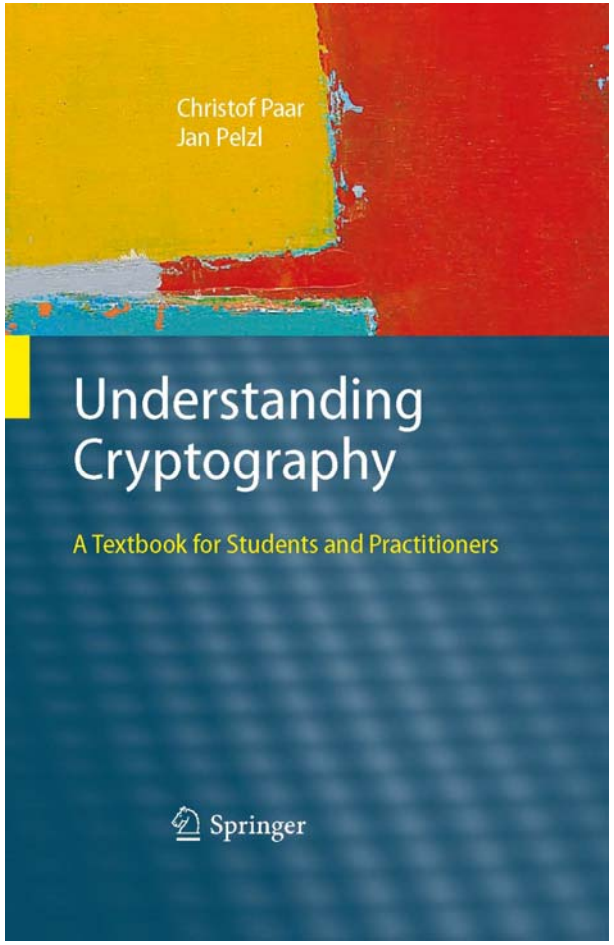
escarUSA – Embedded Security in Cars
Ann Arbor, June 2017



escarEurope – Embedded Security in Cars
Berlin, November 2017

Easy-to-understand book for applied cryptography

RUB



YouTube

[Introduction to Cryptography by Christof Paar]

24 video lectures

Thank you very much for your attention!

Christof Paar

Ruhr Universität Bochum