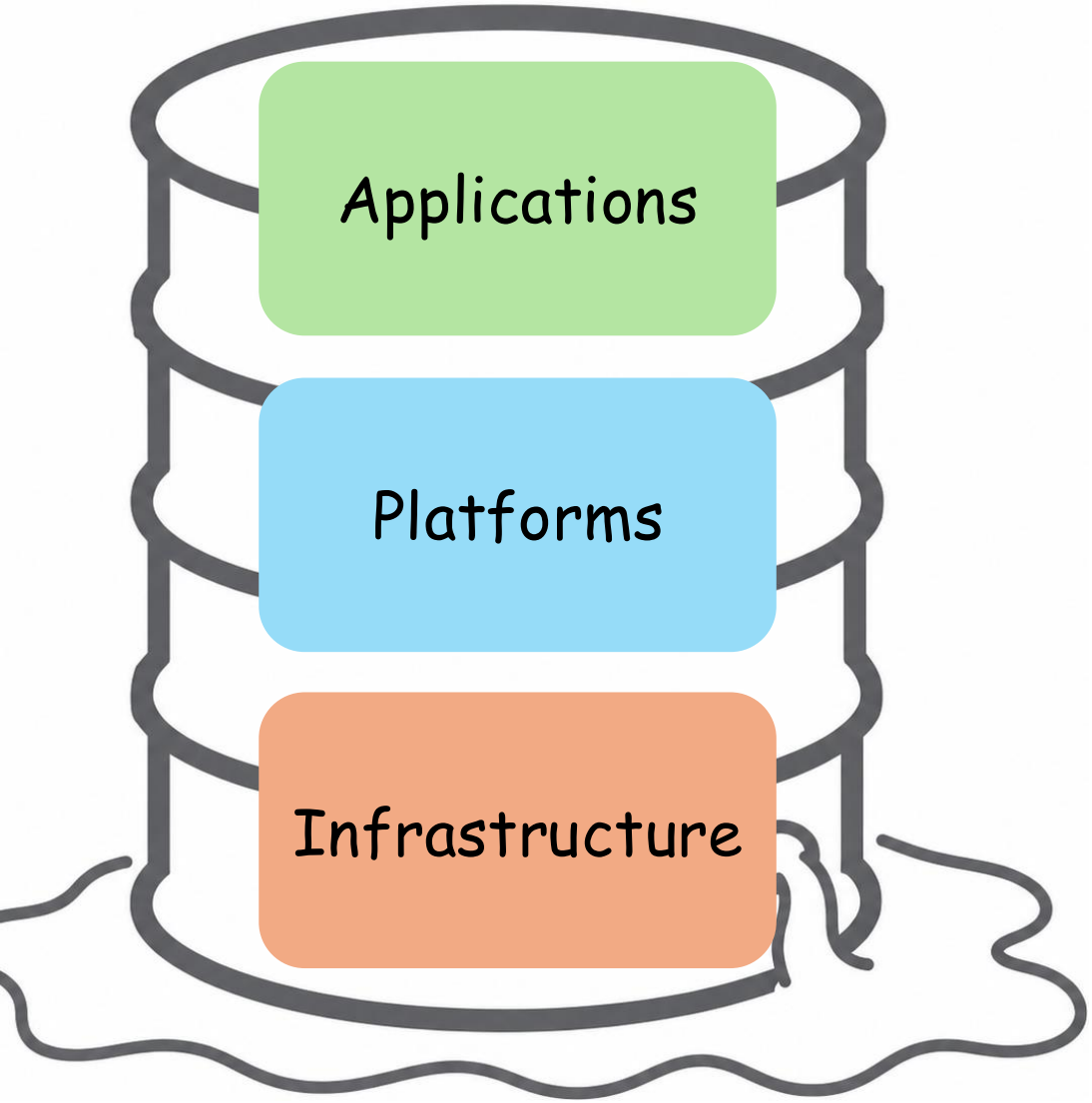
A Primer on Hardware Security

Leon Li, EECS, Washington State University

5/19/2026

Cloud Computing Stack



Services/workflows built on top of platforms

e.g., Quiz, Grading, Discussion, Dashboard, Analytics, Video Gallery, API, LTI integrations.

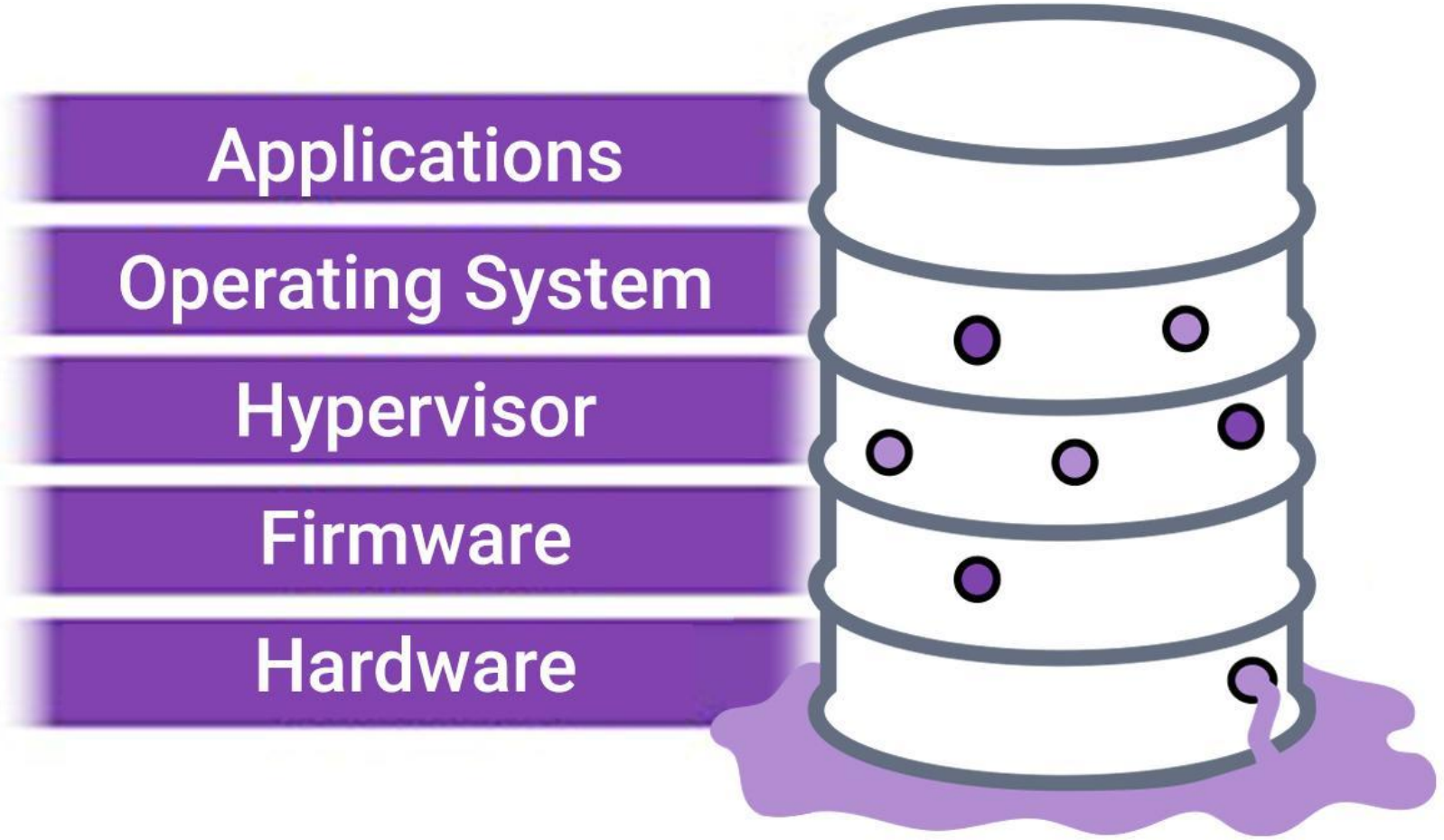
Ecosystems that host/manage applications

e.g., Canvas

Foundational computing resources

e.g., Cloud servers (AWS)

Software Stack



Source: Synopsys

A Primer on Hardware Security: Outline

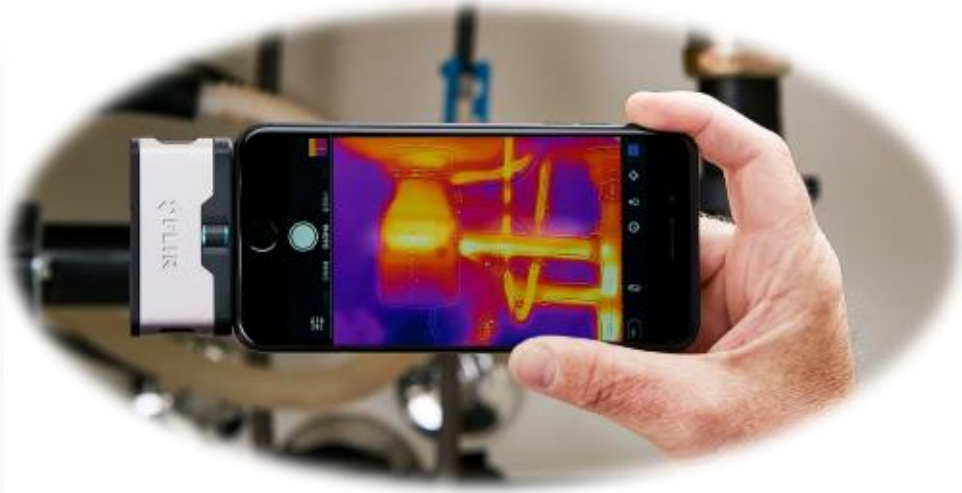
1. Exploiting Leakage from Hardware:
   Side-Channel Attacks
2. Securing Systems with Hardware:
   Hardware Security Modules
3. Protecting Hardware Itself:
   IC Supply Chain Threats & Defenses
4. Towards an Open World:
   Security in a White Box

Exploiting Leakage from Hardware:
Side-Channel Attacks

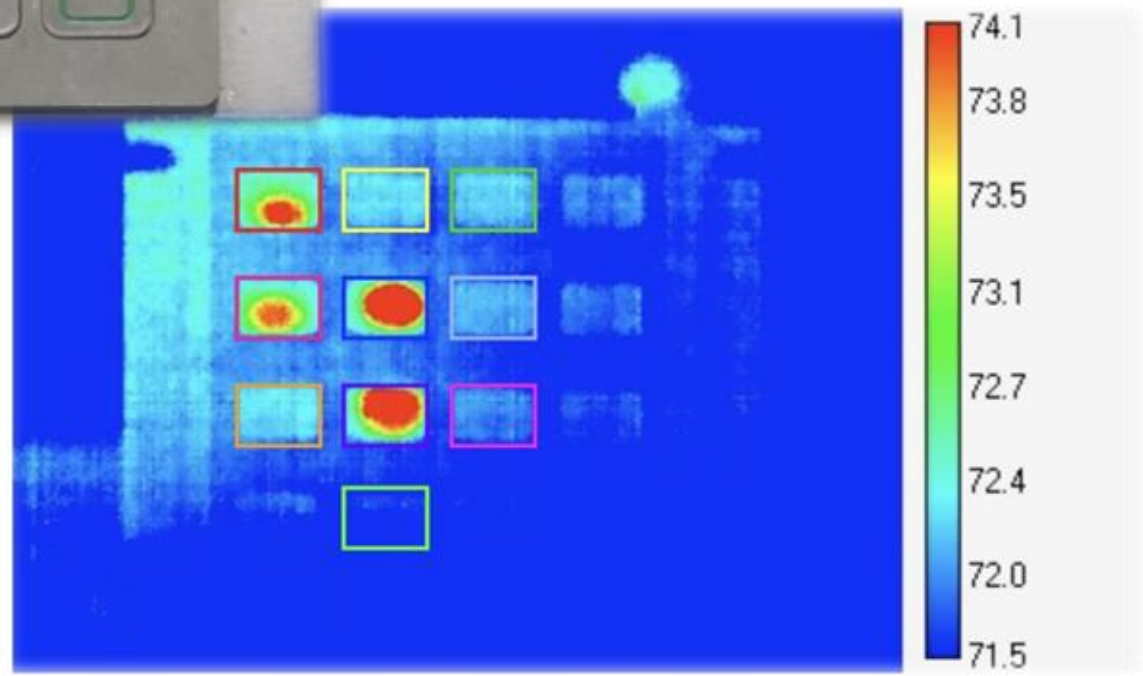
WASHINGTON STATE
UNIVERSITY

Side-Channel Attacks

Thermal side-channel

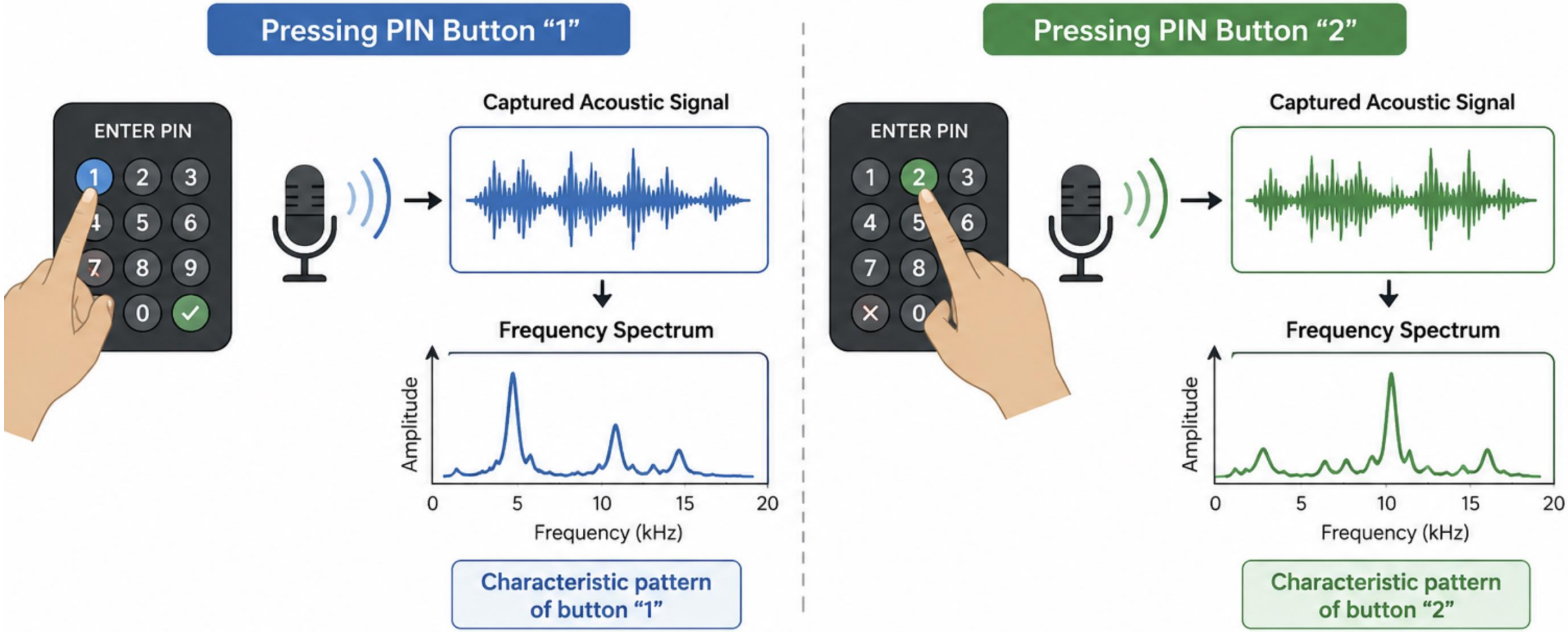


Mechanical side-channel



Side-Channel Attacks

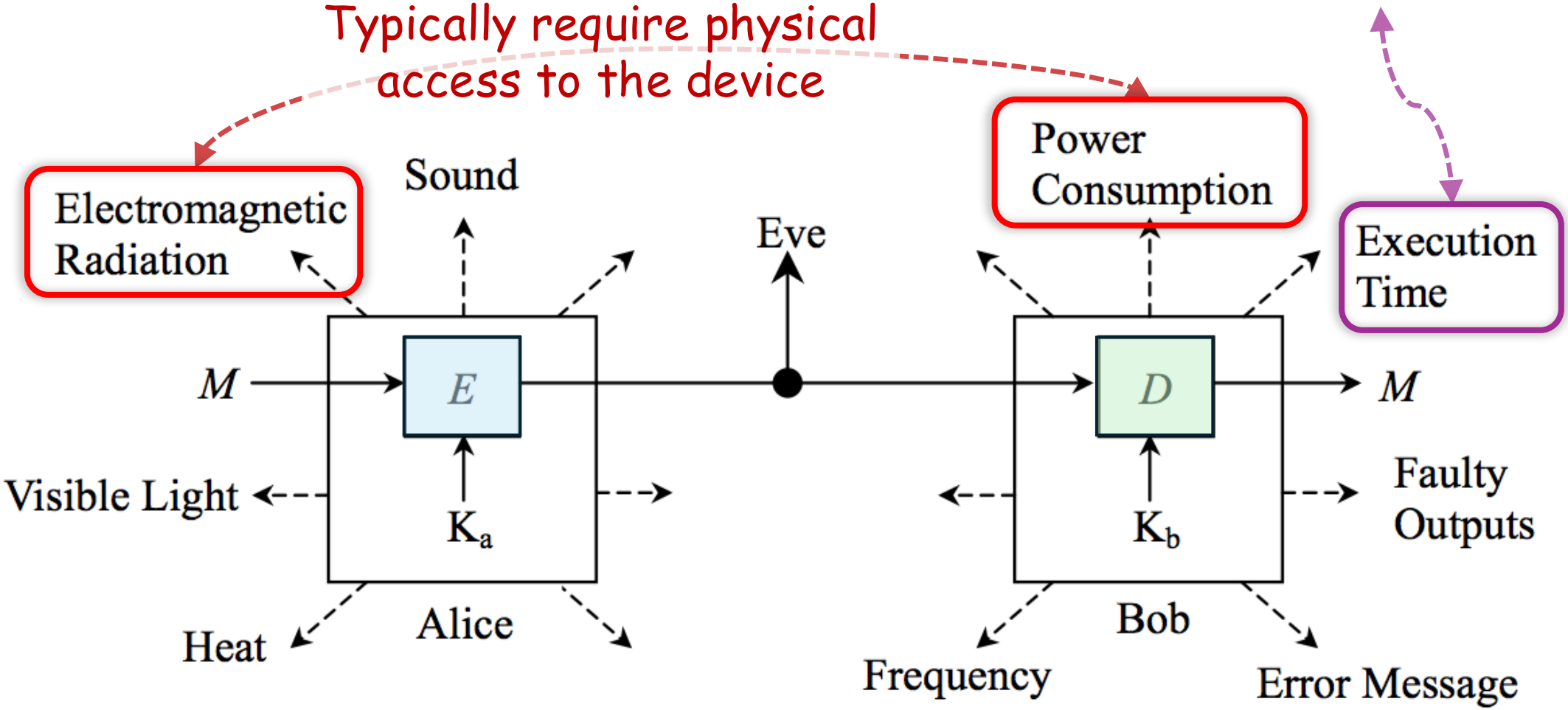
Acoustic side-channel



Side-Channel Attacks

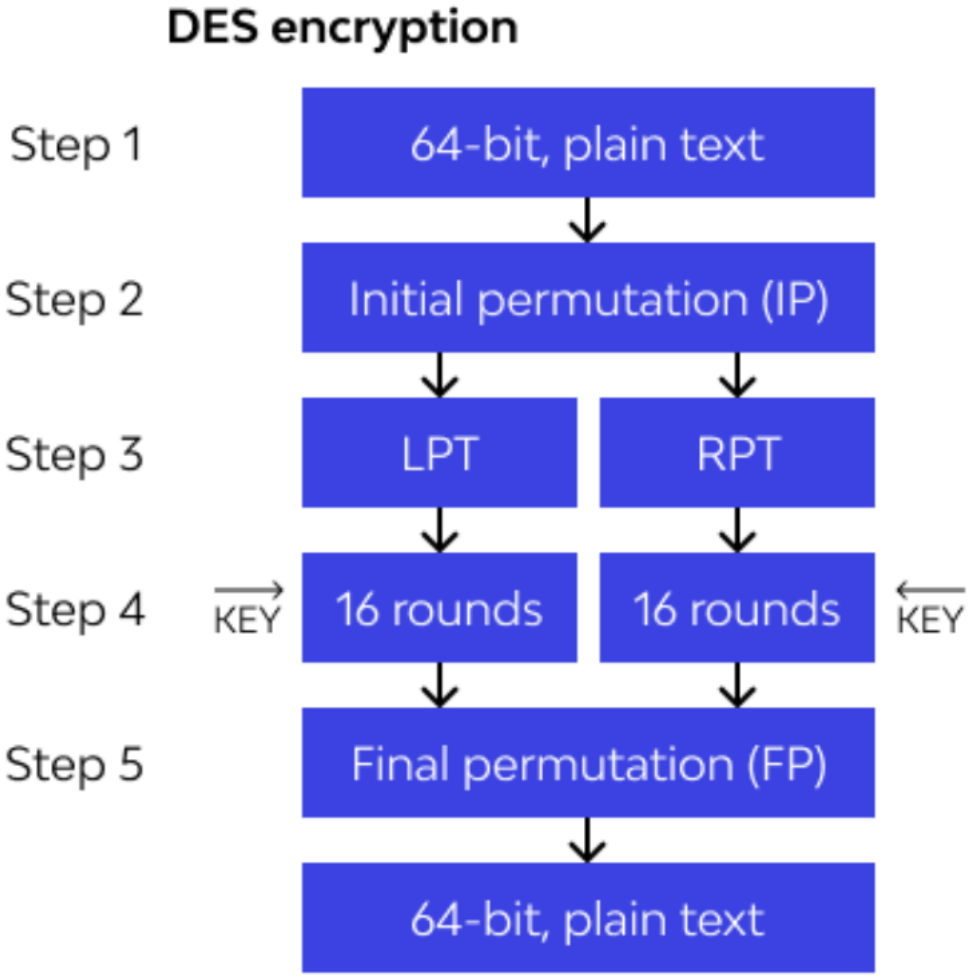
Sometimes measured "indirectly" through remote attacks

Typically require physical access to the device



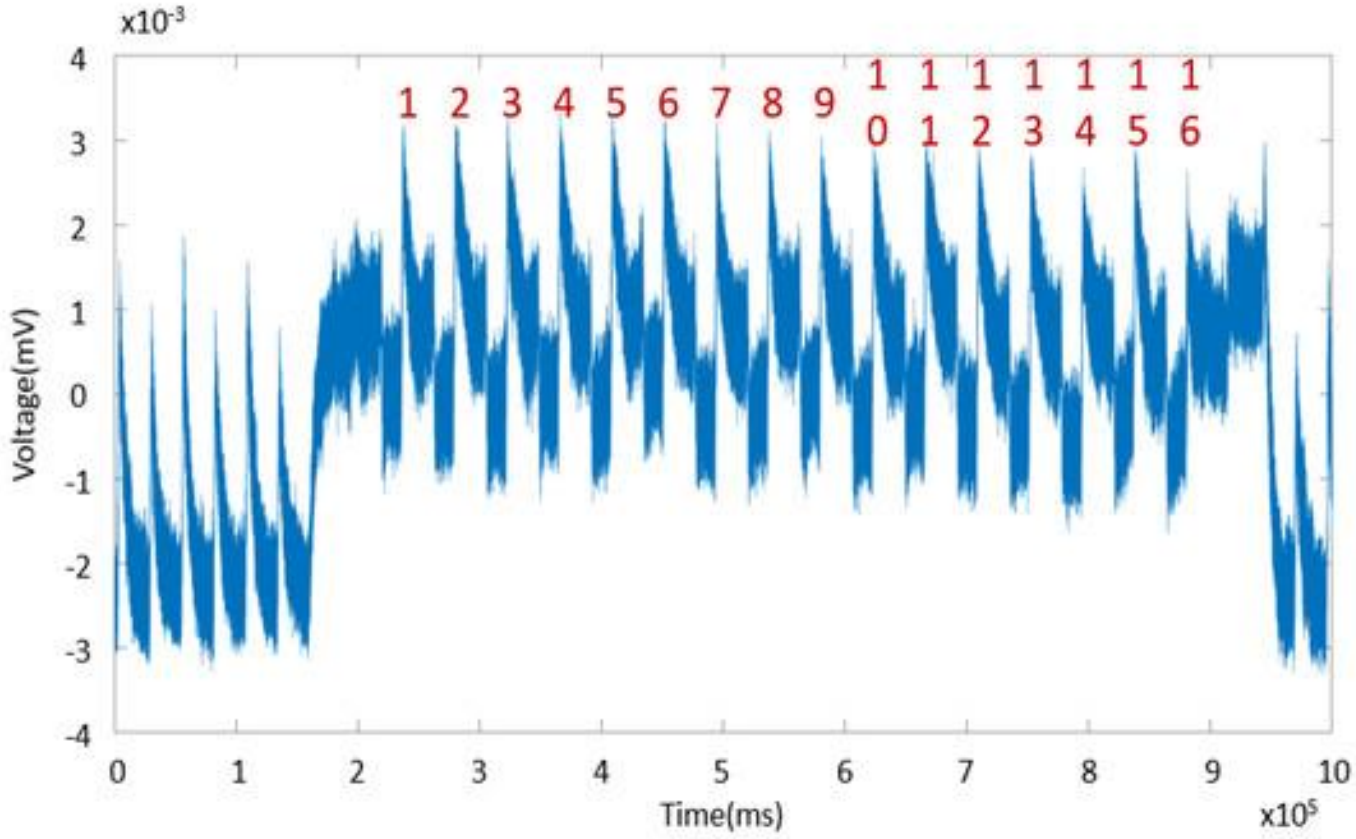
Source: http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-3/physec/papers/physecpaper19.pdf

Simple Power Attacks (SPA)



Source: *wallarm*

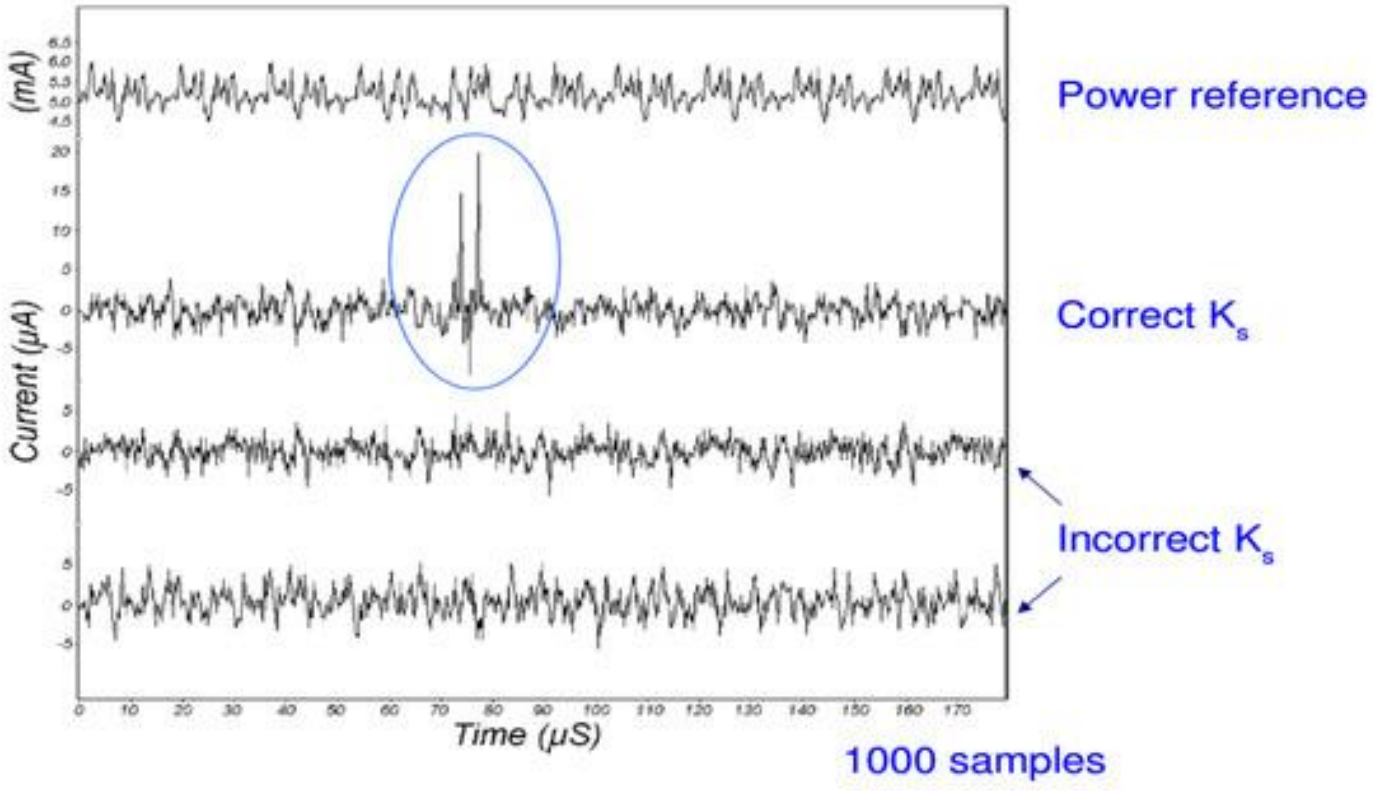
See how power traces align with the sequence of the algorithm



Randolph & Diehl, *Cryptography*, 2020

Differential Power Attacks (DPA)

- **Collect a lot of traces**
- Divide traces into 2 groups based on internal behaviors under a **key guess**
- If the 2 groups exhibit **substantial differences:**
  - The *predicate* is valid
  - Thus, the key guess is correct

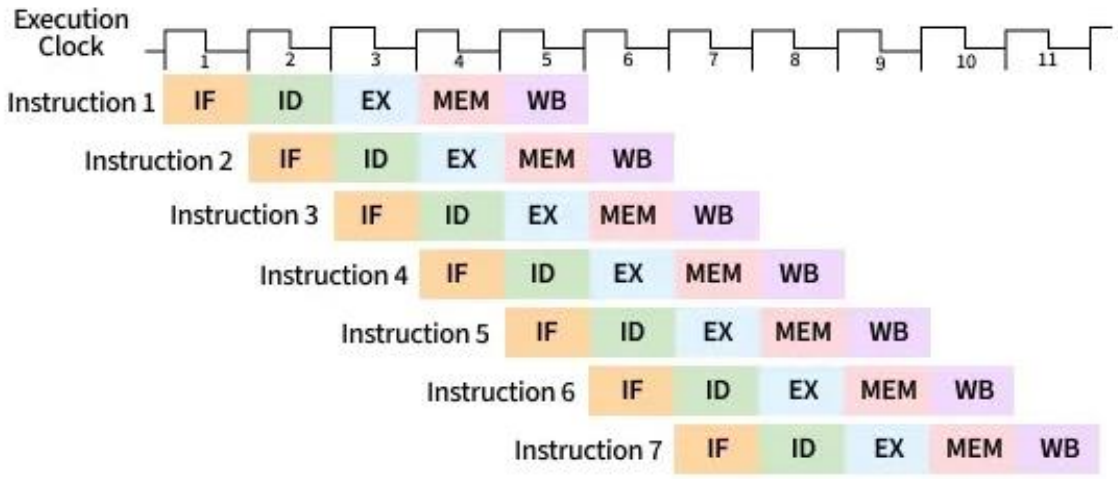


Kocher, P.; Jaffe, J.; Jun, B. Differential power analysis. 1999

Side-Channel Attacks on CPUs

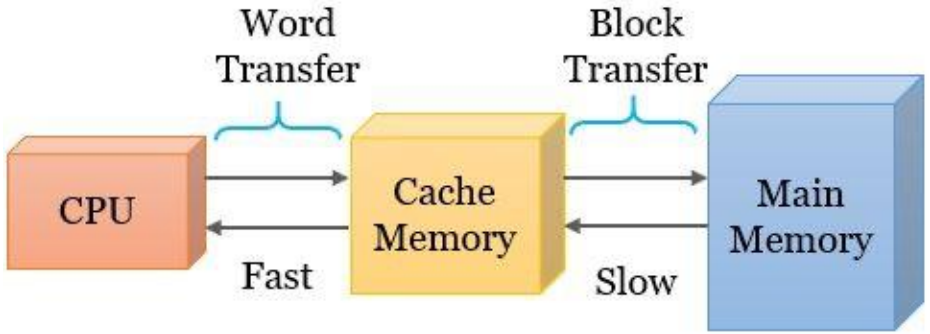
- Modern processors are packed with “tricks” to run faster

Pipeline



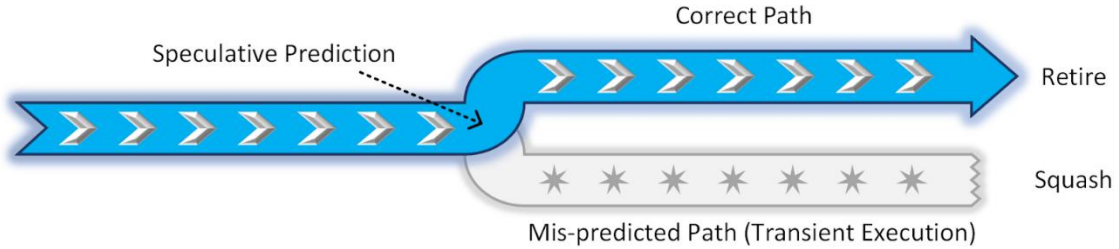
Source: GeeksforGeeks

Cache



Source: Tech Differences

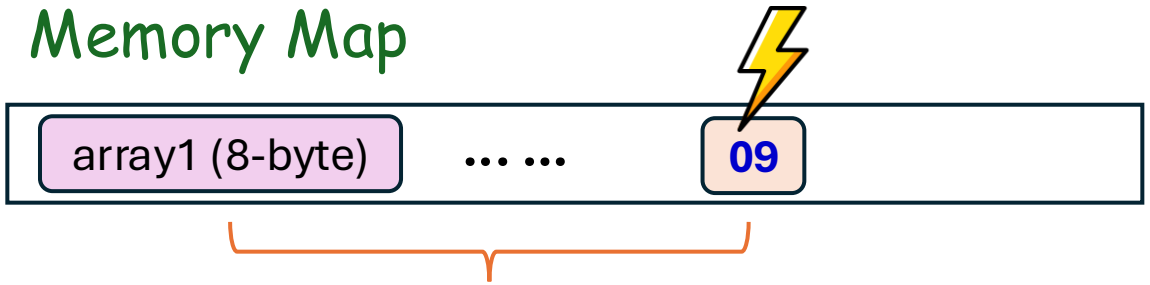
Speculative Execution



Source: Intel

Spectre Attacks on CPUs

Memory Map



Code

```
if (x < array1_size)
    y = array2 [array1 [x] * 512]
```

- Code runs in a trusted context
- Adversary wants to read memory and controls unsigned integer x

Attack Process

- Attacker calls victim code with `x=80`
- Predict if() is true
- Read at `array1` base+80, returns byte = **09**
- Request memory at (`array2` base + 09*512)
- Brings `array2[09*512]` into cache
- Realize if() is false: discard speculative work

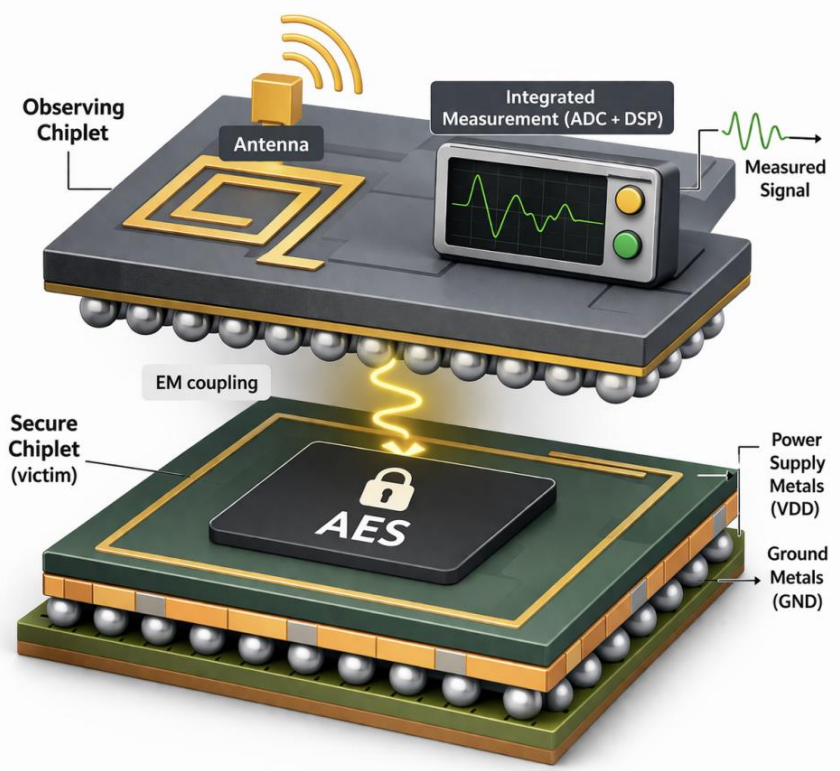
- Attacker times reads from `array2[i*512]`
- `i=01` is slow, `i=02` is slow, …. `i=09` is fast
  - → secret = **09**!

Key Takeaways

- Power side-channel attacks happen because the encryption process manipulates data-dependent intermediate values, causing measurable differences in **power consumption, timing, electromagnetic radiation, or cache activity** during execution.
  - Secure algorithm ≠ Secure implementation
- Modern CPUs became vulnerable to Spectre because they aggressively optimize for **performance** using speculation, caching, and prediction.
  - Performance optimization → observable physical behavior
- Security is not just a technical problem — it is also an **economic and ecosystem problem** involving vendors, users, incentives, and standards.

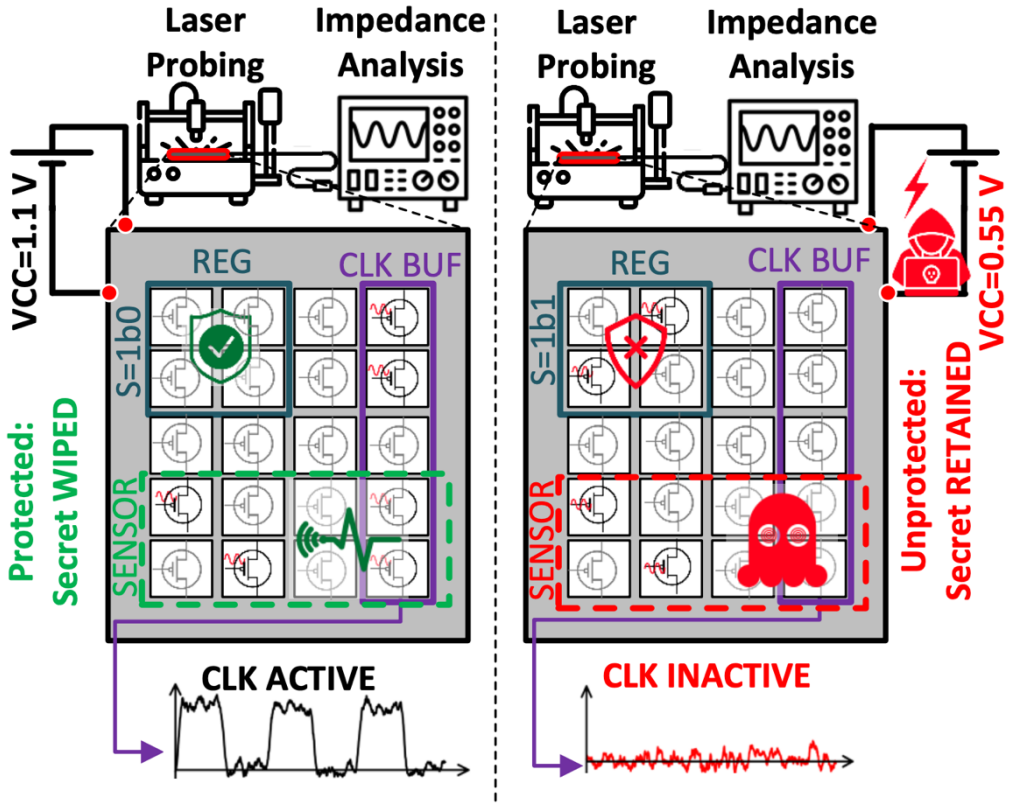
Emerging Frontier of Side-Channel Attacks

Emerging chip design technologies (e.g., chiplet) provide novel access channels



Source: Spying Across Chiplets

More sophisticated control and observation mechanisms



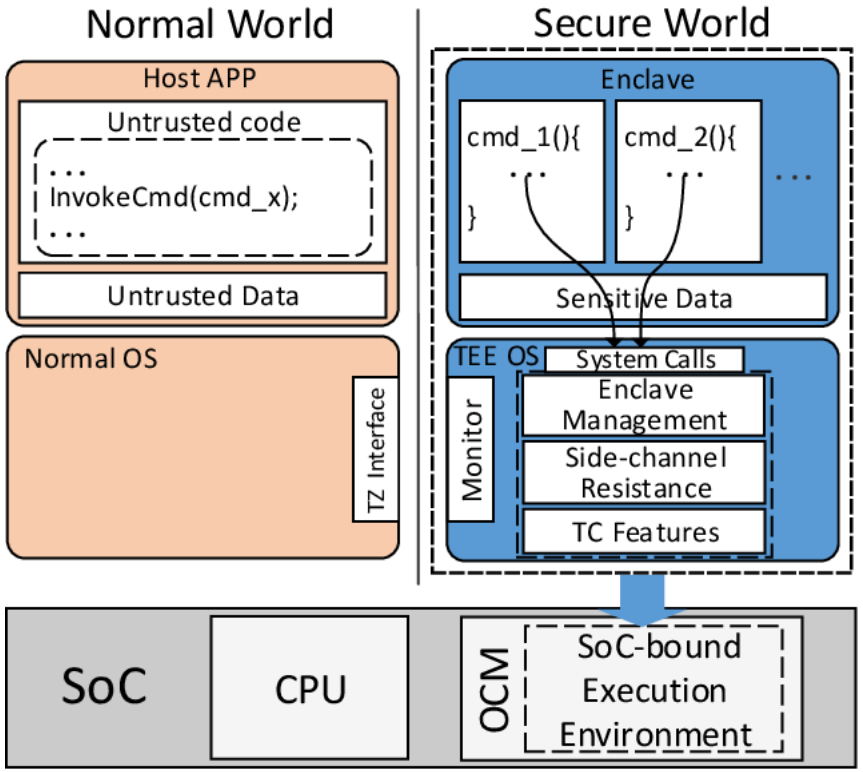
Source: Chypnosis

Securing Systems with Hardware:
Hardware Security Modules

WASHINGTON STATE
UNIVERSITY

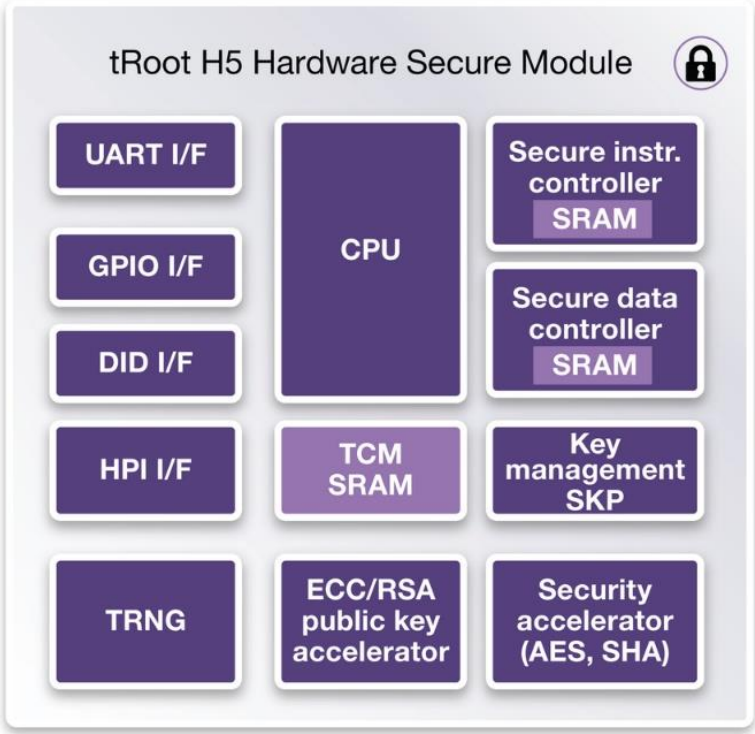
System Security Requirements

We want a **separation** between "Normal World" & "Secure World"



Source: SecTEE

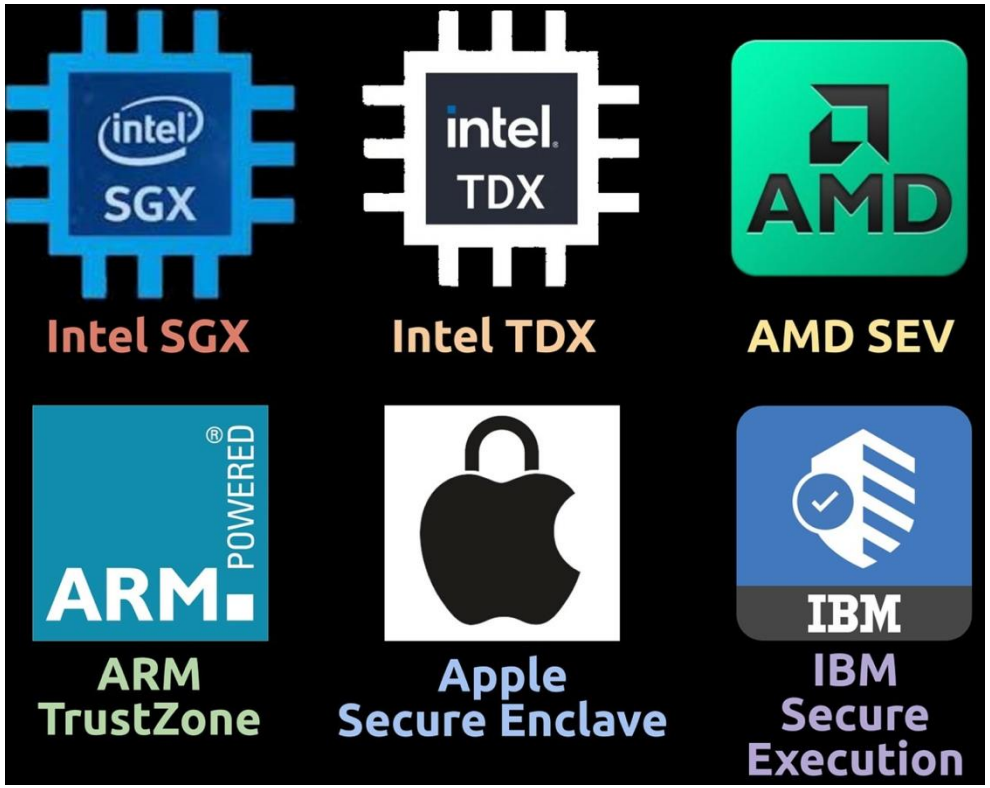
We need hardware support for **primitive** security capabilities



Source: Semiconductor Engineering

Hardware Security Modules

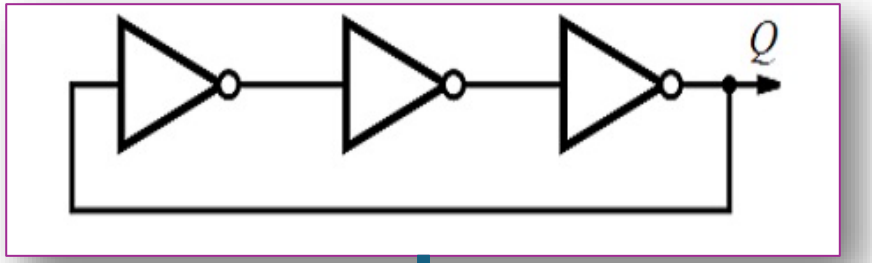
- A **secure area** within the main processor that ensures code and data loaded inside are protected with respect to **confidentiality and integrity**.
  - Cryptographic Acceleration
  - True Random Number Generation
  - Secure Storage
  - Isolated Secure Execution
  - Secure Boot and Attestation
  - … …



Source: @dan_nanni

Exploitation of Randomness

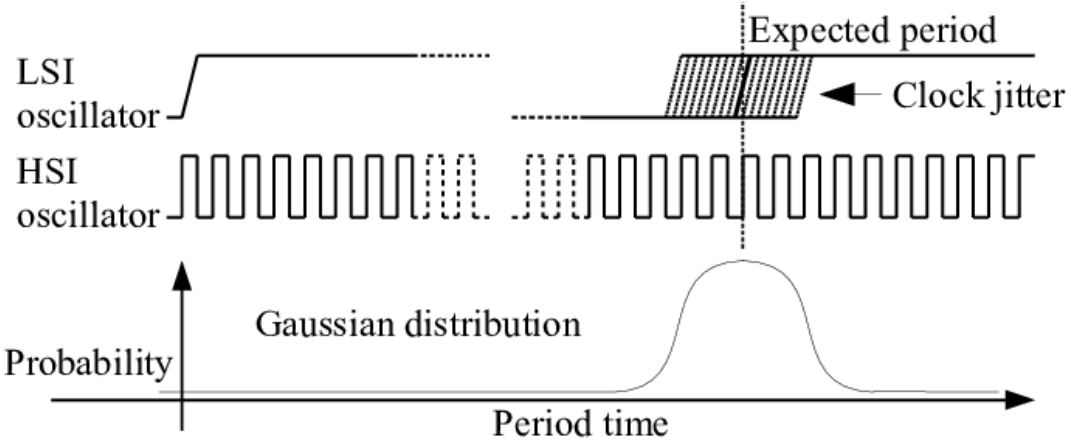
Ring Oscillator is a great entropy source

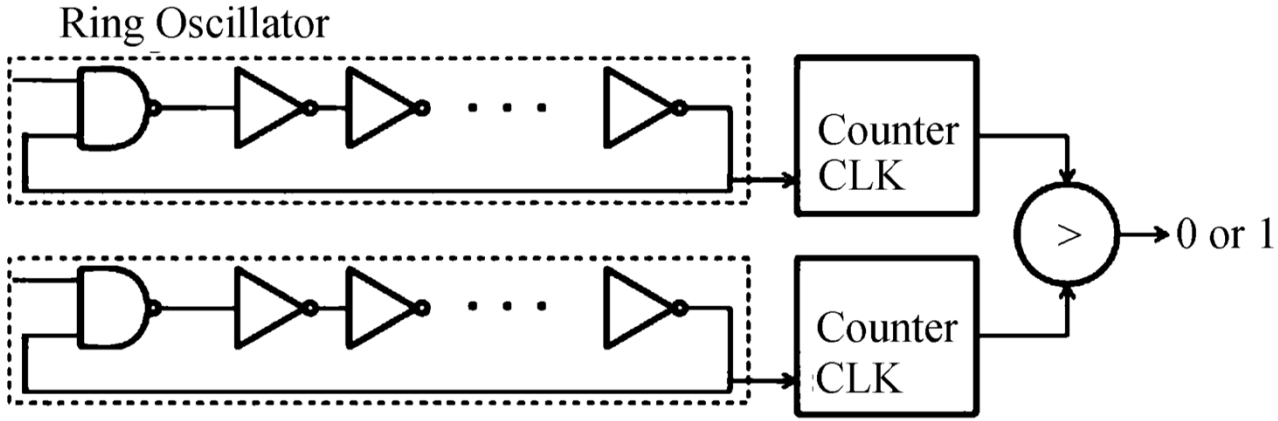


0 → 1 → 0 → 1 → …

True Random Number Generation with **Clock Jitter**

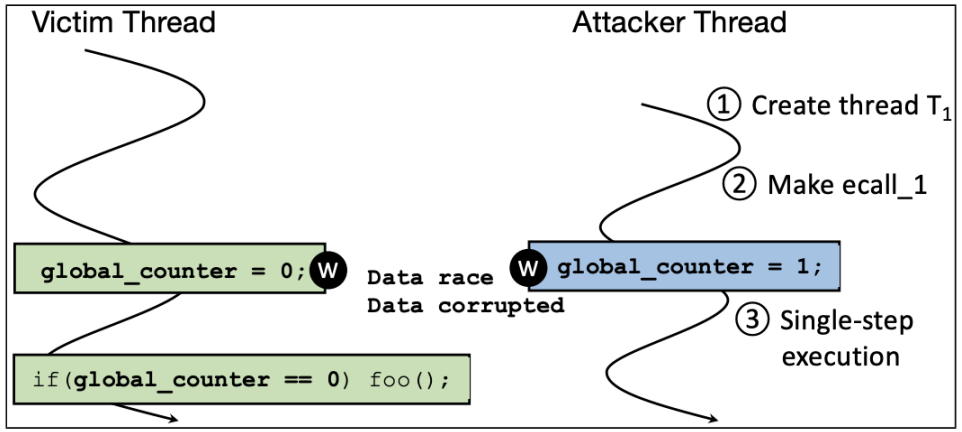
Physical Unclonable Function (PUF) enables unique device identification





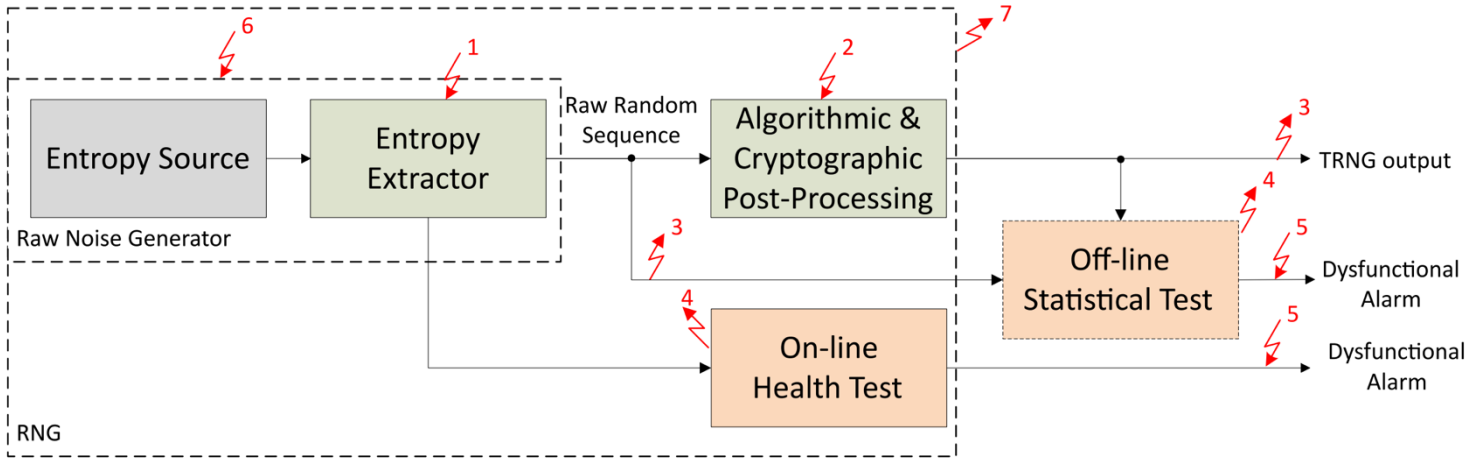
Open Questions

The **global interactions** still leave opportunities for exploitations



Chen, Sanchuan, Zhiqiang Lin, and Yinqian Zhang. "Controlled data races in enclaves: Attacks and detection." USENIX 2023.

Fault-injection and **aging** can compromise randomness



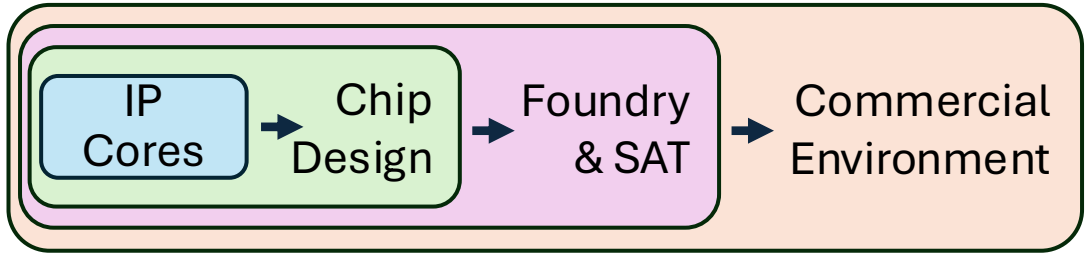
Yu, Yang, Michail Moraitis, and Elena Dubrova. "Can deep learning break a true random number generator?." *IEEE Transactions on Circuits and Systems II: Express Briefs* 68.5 (2021): 1710-1714.

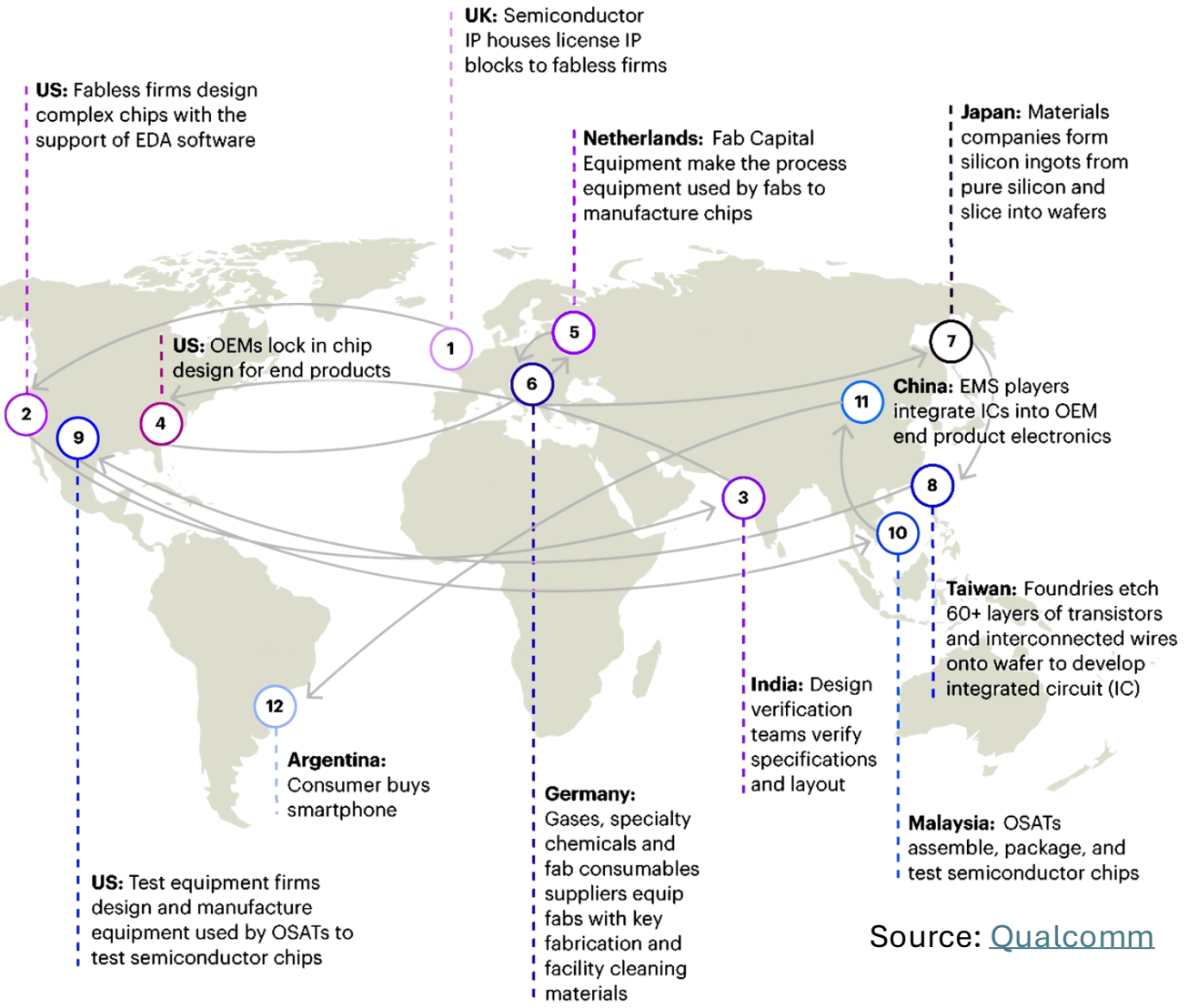
Protecting Hardware Itself:
IC Supply Chain Threats

WASHINGTON STATE
UNIVERSITY

Globalization of Integrated Circuit (IC) Industry

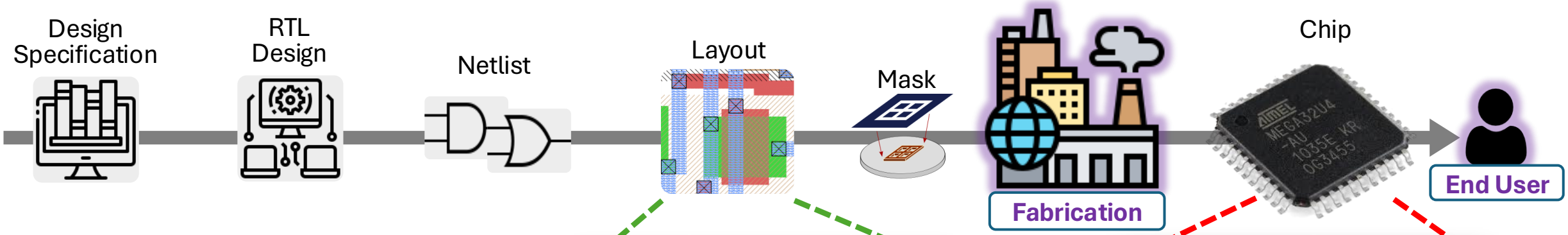
- The industry has evolved into a deeply **interconnected web of global partners.**
- Constant movement of intellectual products across trust boundaries:
  - IP vendor → chip designer
  - chip designer → foundry/SAT
  - production → market



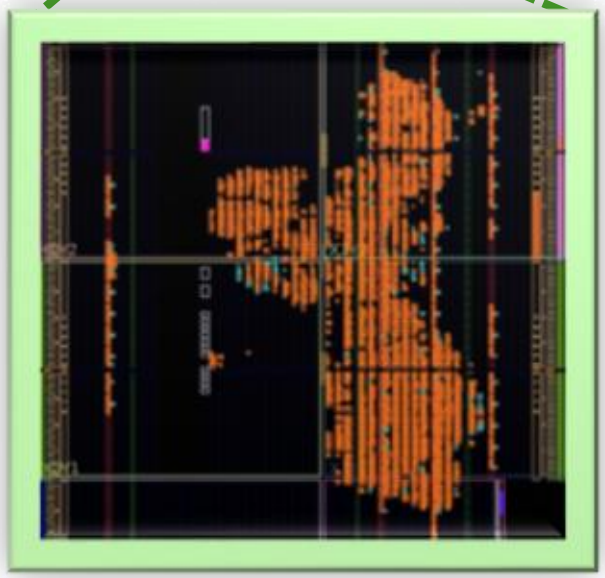


Source: Qualcomm

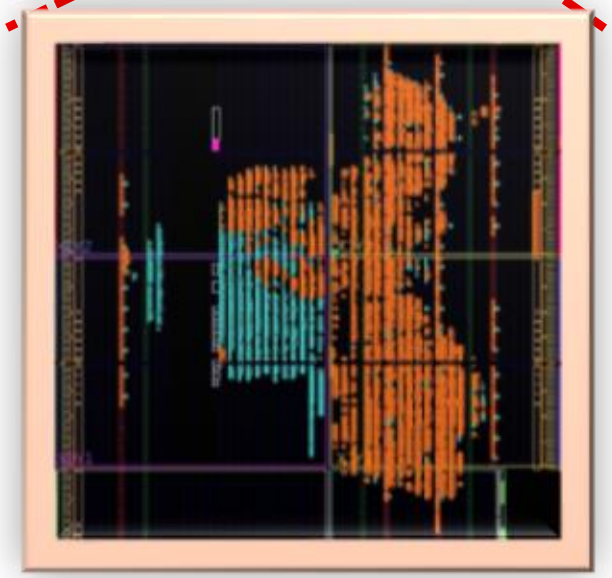
Hardware Trojans



- **Adversary (malicious foundry)** modifies the IC design by embedding backdoors



Correct Design

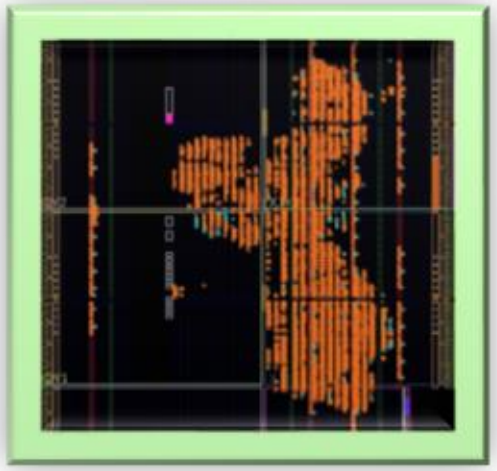


Trojan-infected Design

Hardware Trojan Detection

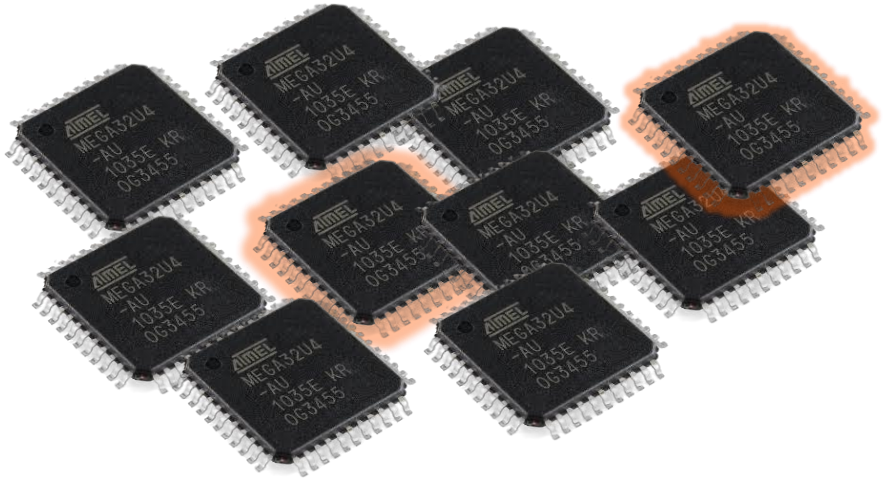
- Trojan detection has been challenging
  - We don't get to see the internals of every fabricated chip.
  - Trojan is **very covert** and has little impact on functional and side-channel behaviors
  - Trojan footprint is **tiny** and can hardly be located

What We Asked

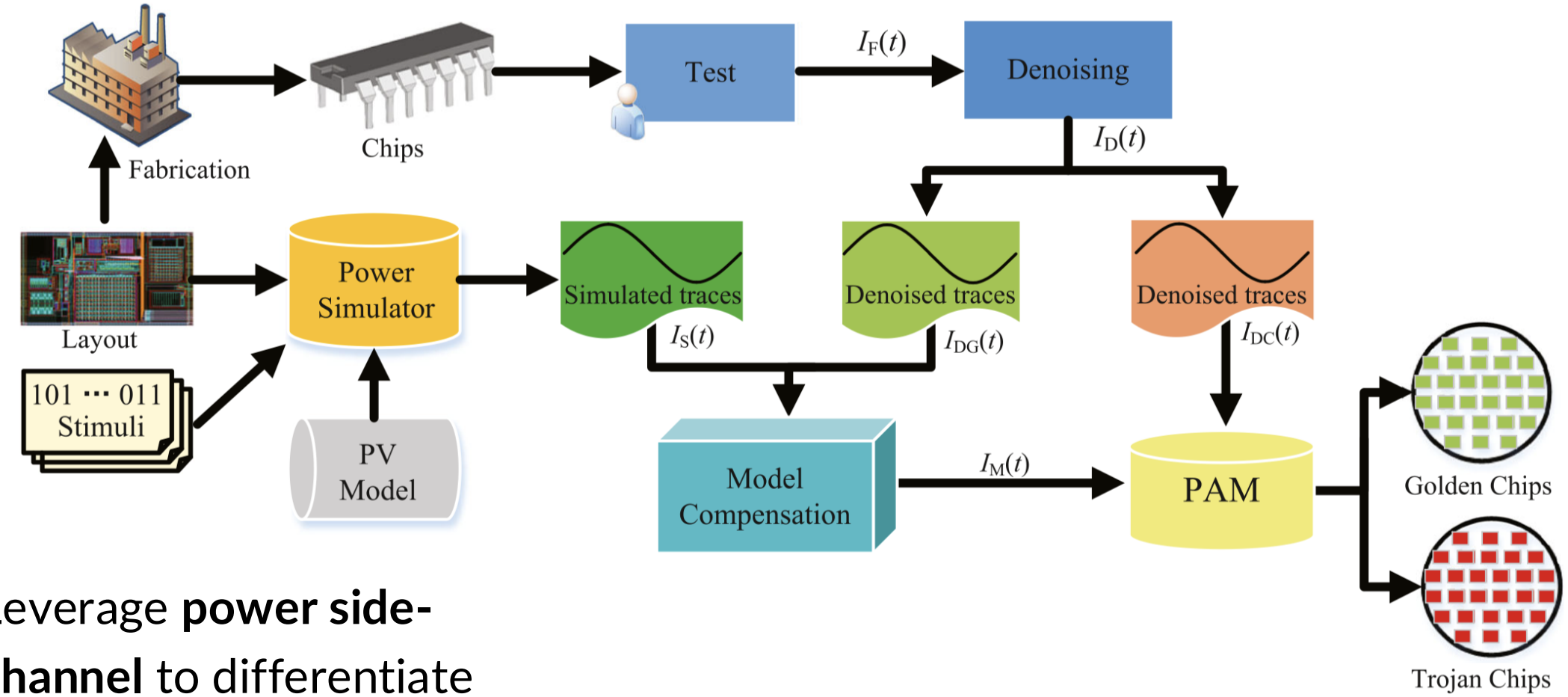


How do We Compare?

What We Got



Hardware Trojan Detection

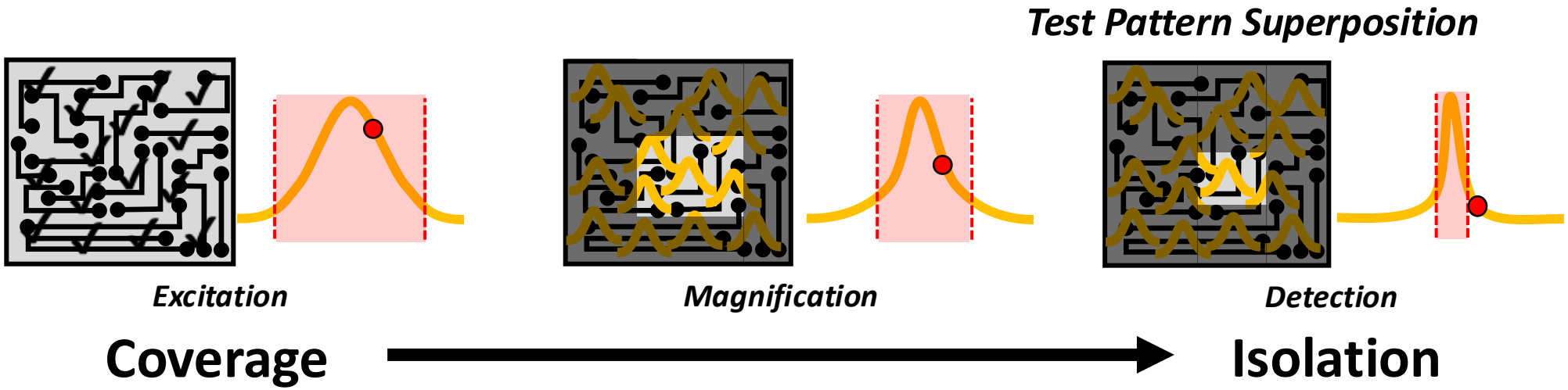


- Leverage **power side-channel** to differentiate infected chips

Liu, Yanjiang, et al. "Hardware trojan detection leveraging a novel golden layout model towards practical applications." Journal of Electronic Testing 35.4 (2019): 529-541.

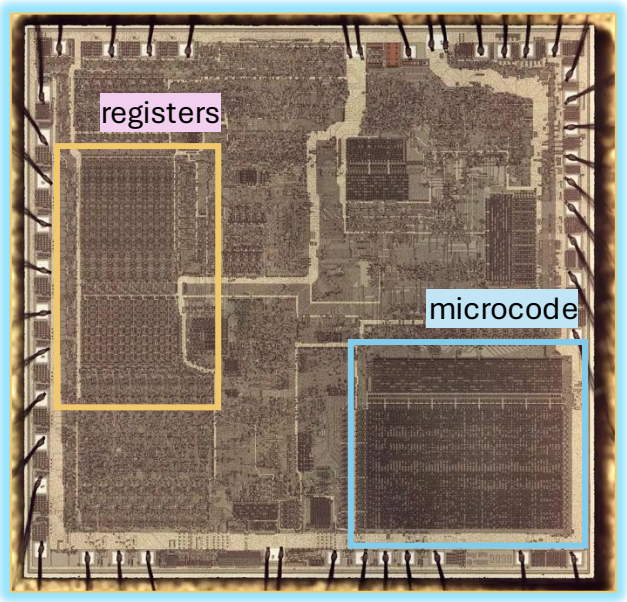
Hardware Trojan Detection

- What **stimuli/patterns** do we use for simulation/testing?
  - Generate high-coverage test patterns to ensure Trojan logic is activated
  - **Refine test patterns using *superposition* techniques** to magnify the impact of Trojans
  - Achieved more than 300X improvements in revealing Trojan signals

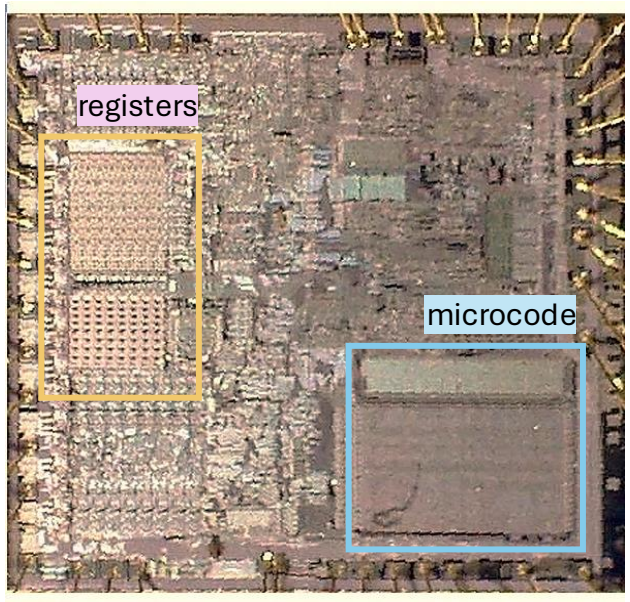


Piracy and Reverse Engineering

Original
Intel 8086

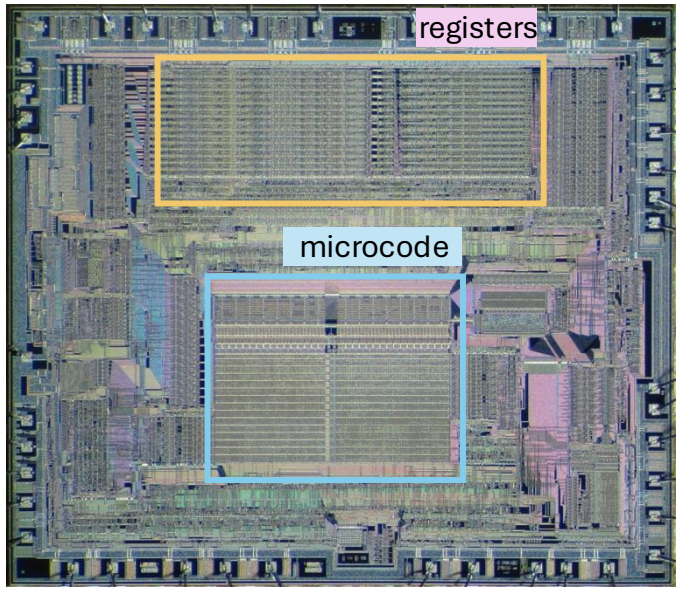


Direct photographic clone
Soviet K1810VM86 [1]



- Identical layout
- Slightly **slower**

Enhanced design from Reverse Engineering
NEC V30 [2]

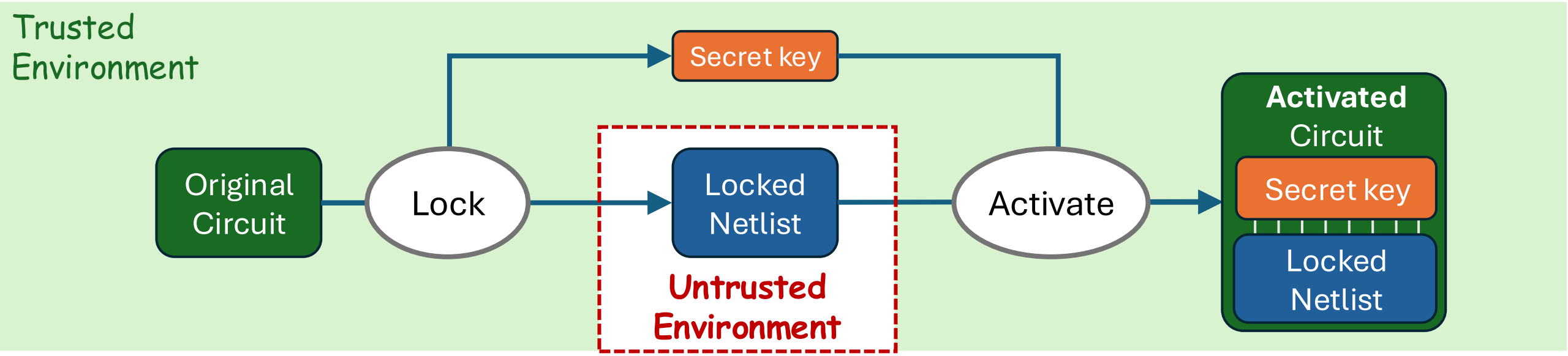


- New microarchitecture based on **reverse-engineered** microcode
- Pin-compatible and substantially **faster** (10%-40%)

[1] https://habr.com/ru/articles/493294/comments/
[2] https://en.m.wikipedia.org/wiki/File:NEC_V30_die.JPG

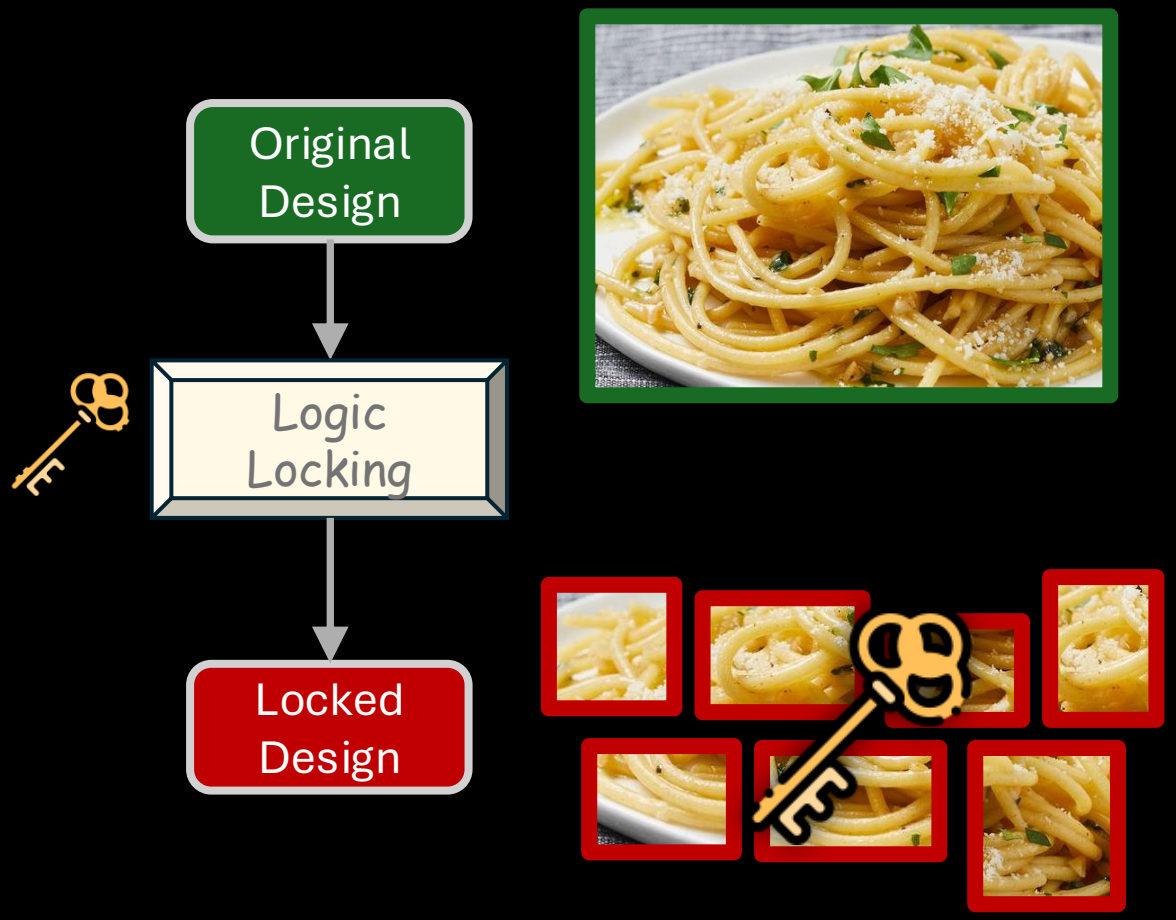
Logic Locking

- A **design-for-trust** technique to shield against piracy, overproduction, and RE.
- Lock the design before sending it to untrusted environments.
- The key is loaded in a Tamper-Proof Memory (TPM) through a trustworthy channel.



Logic Locking – Pasta Recipe Analogy

Locking Process



Unlocking Possibilities

Key 1

Key 2

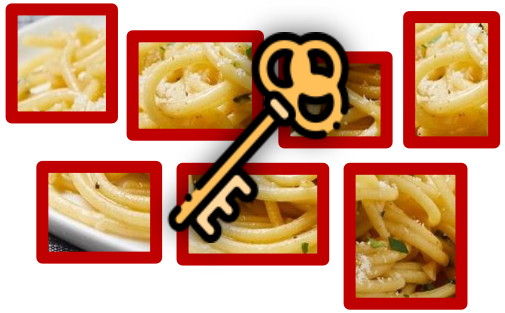
Key 3

Key 4

Weakness of Logic Locking

Functional Weakness

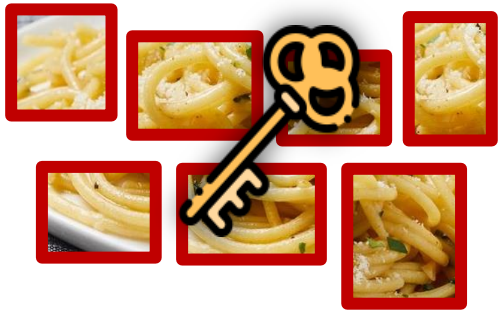
"Key Recover by Tastes"



+ Blind tasting of the expected dish
(i.e., oracle query)

Implementation Weakness

"Key Recover by Split Patterns"



+ Understanding of how the cooking & hiding processes work
(i.e., locking & synthesis tool)

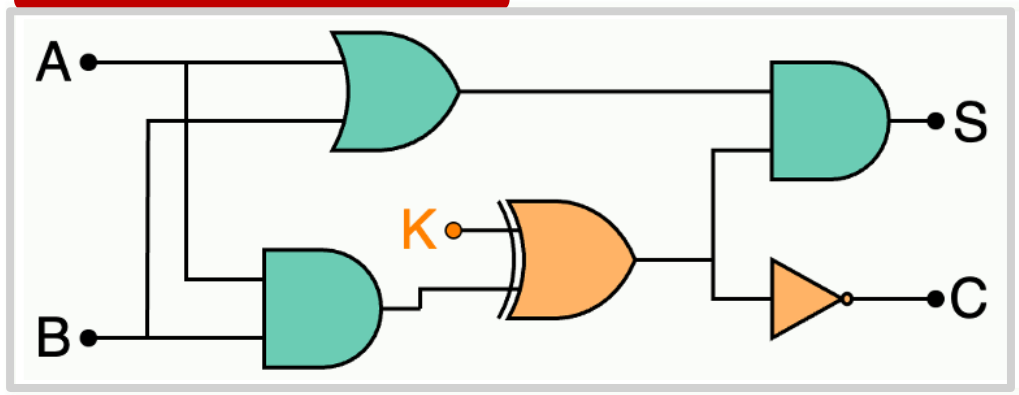
Structural Weakness

"Key Recover by Looks"

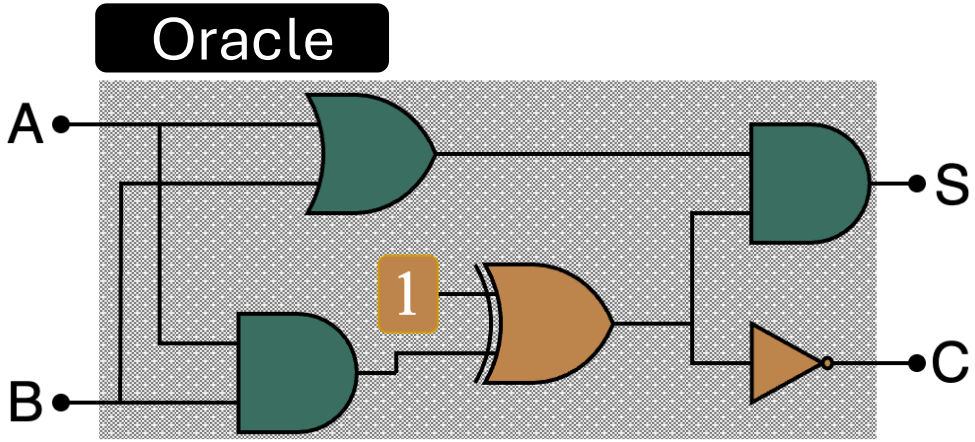
+ Nothing!

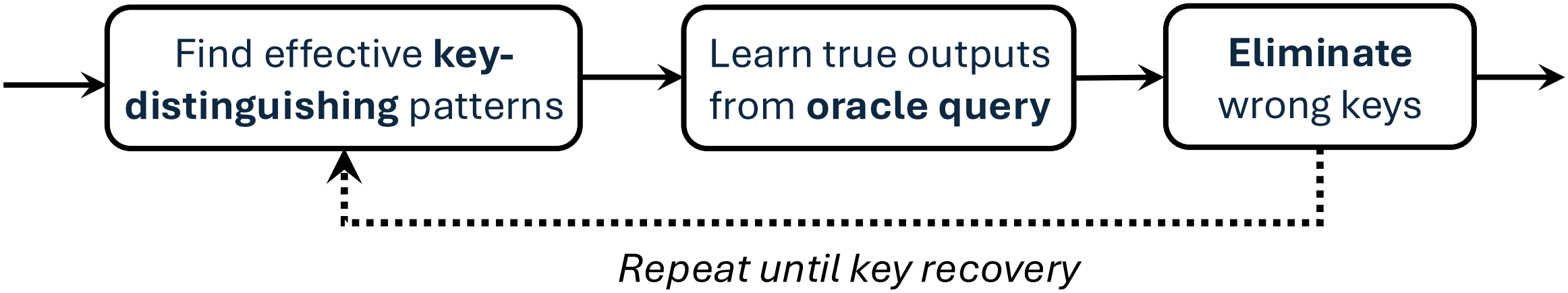
Functional Weakness of Logic Locking

Locked Netlist



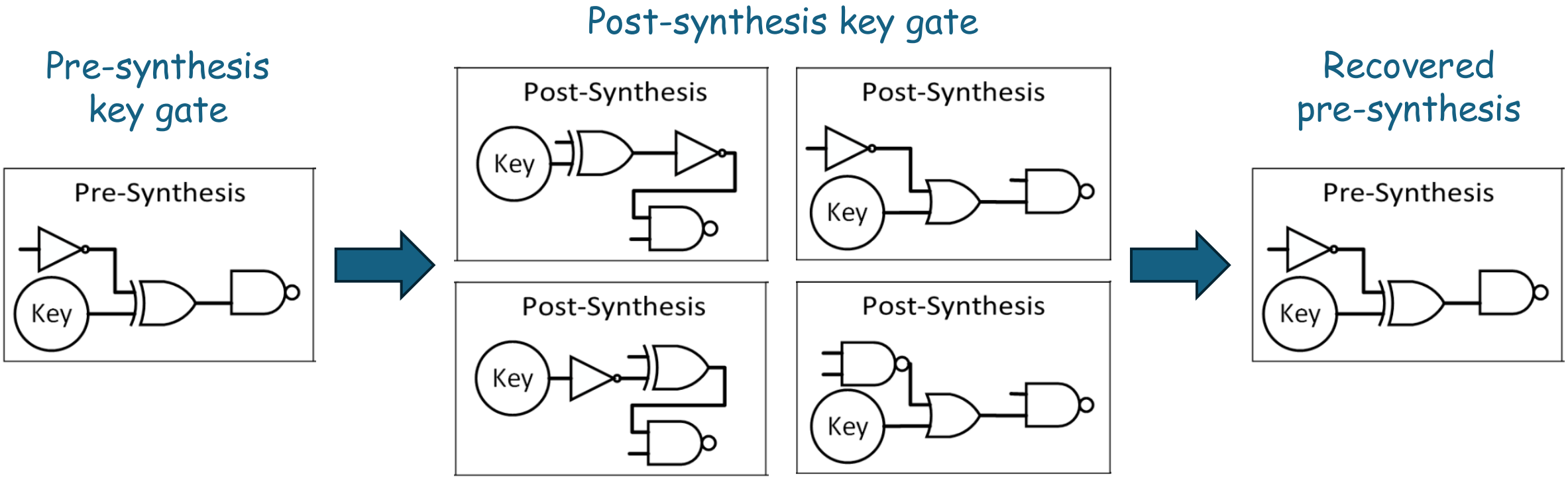
Oracle





Implementation Weakness

- Locking tools exhibit a **largely deterministic and predictable behavior**
- ML model can be trained to learn these behaviors and **reverse** the process



Chakraborty, Prabuddha, et,al. "SAIL: Machine learning guided structural analysis attack on hardware obfuscation." 2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST). IEEE, 2018.

Structural Weakness

As long as we know it's supposed to be a rational Italian dish...

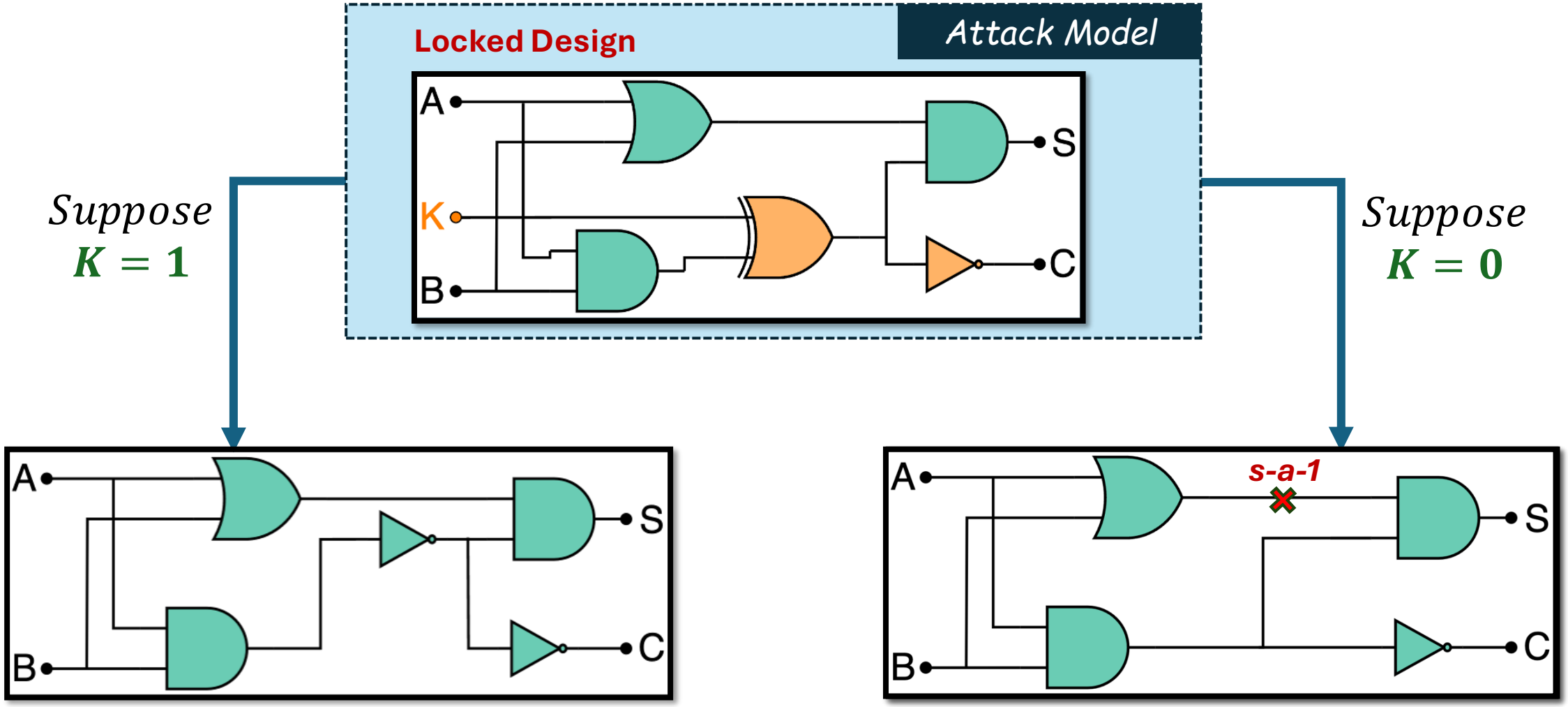
Ketchup is a no-no

Who would snap spaghetti in half?

This looks delicious!

PINEAPPLE in PASTA?

Structural Weakness



✅ The circuit is irredundant

❌ Redundant logic exists

Vicious Cycles of Logic Locking

- Functional and structural traces lead to continuous vulnerability discovery
- Derive internal secrets from the original design to deliver corruption

Internal Secrets
Only the designer can calculate

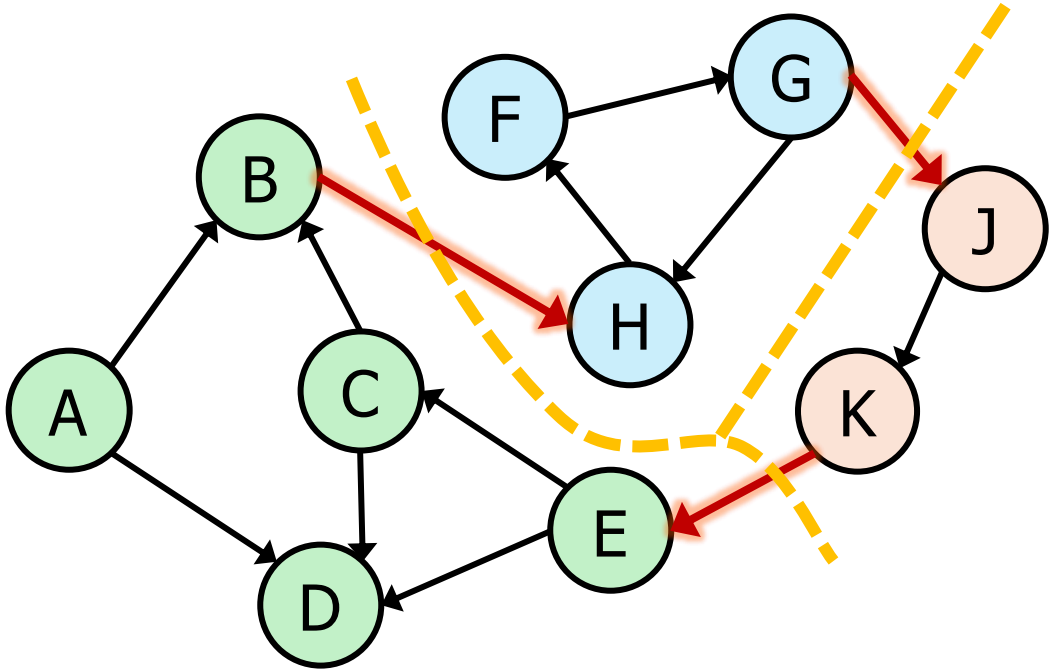
Original Design
Only the designer can access

Corruption
Protection against hardware reverse engineering

External Secrets
Only the designer has access to

Key Exposure
Clues towards key recovery

Components of JANUS Obfuscation



A/ Configuration-based Locking

Synthesize the design under configurations to produce a corrupted netlist.

B/ Configuration Assignment

Determine the configurations that minimize key exposure

C/ Hybrid Activation Control

Combine intrinsic sequential evoluation & external secret key for activation control

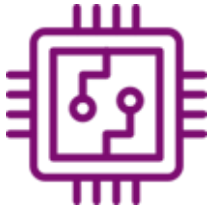
Towards an Open World:
Security in a White Box

Threat Landscape

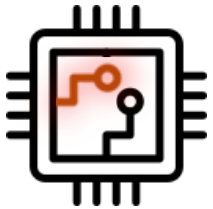
Attacker's Objective

Attacker's Visibility

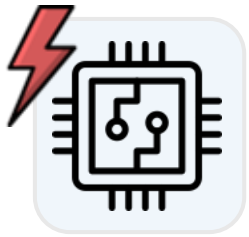
Counterfeiting & piracy



Malicious modification



Fault injection



IO enumeration -based cloning

Competitor RE

Operating Condition

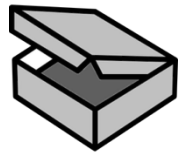
Disruption &

Denial-of-Service

Foundry overproduction/ IP core overuse

Stealthy & Precise Payload

Black-box



Gray-box

White-box

Threat Landscape

Avenues of gaining information

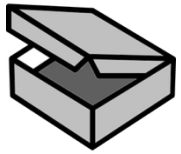
Design
Sharing

directly expands
adversarial access

Reverse
Engineering

enables adversarial
access escalation

Black-box



Gray-box

White-box

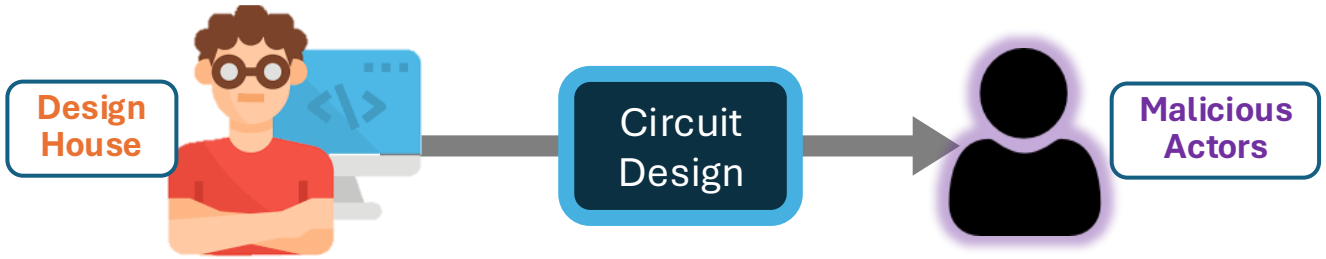
Threat levels

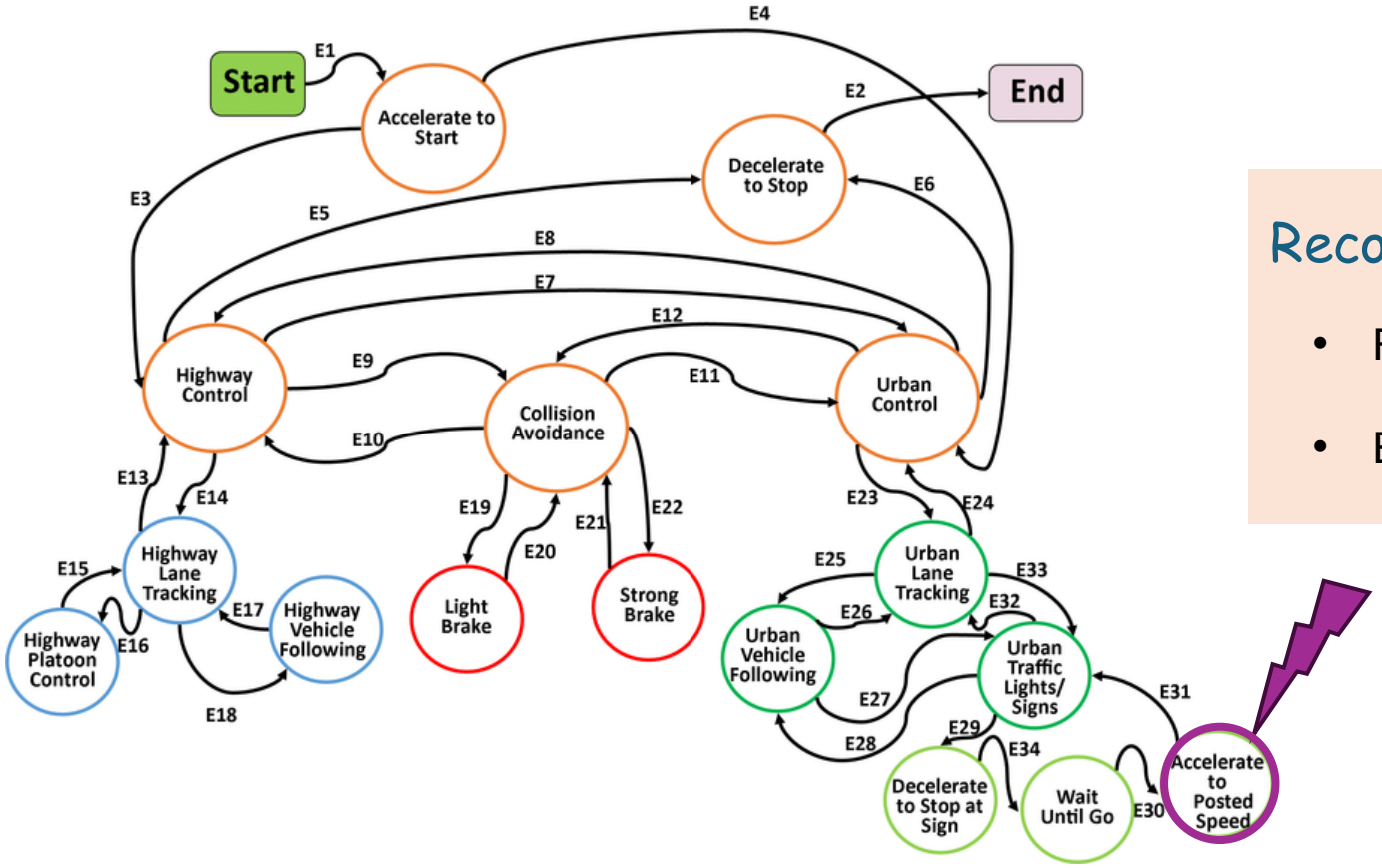
More visibility
for attackers

=

More danger
for IP right owner

Functional Reverse Engineering

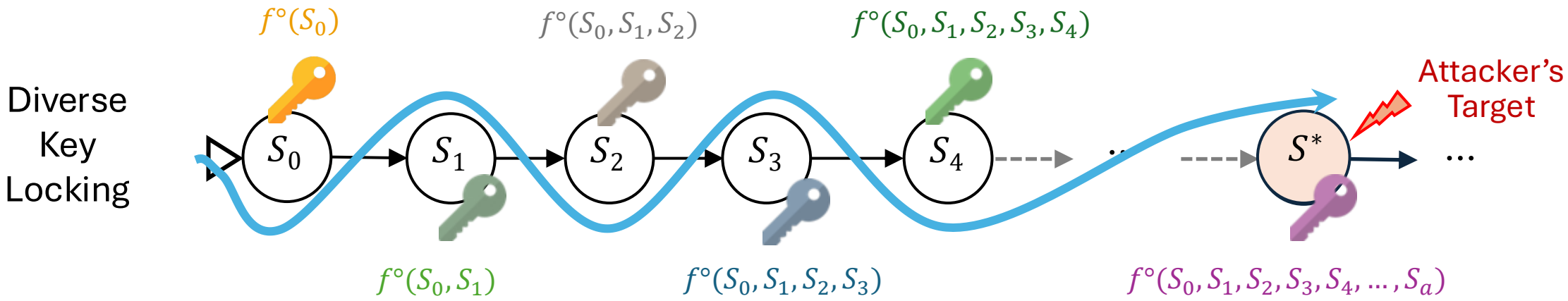




Recover functionality at chosen conditions

- Foundation for all subsequent functional analyses
- Easily accomplished via combinational simulation

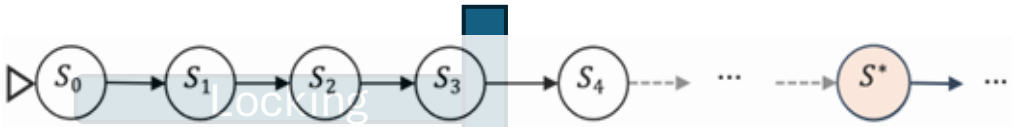
White-box Logic Obfuscation (WBLO)



Design Time

Key Assignment

$Key(S^*) = f°(S_0, S_1, S_2, \dots\dots, S^*)$



$Out^{locked}(S^*, X) = Out(S^*, X) \oplus Key(S^*)$

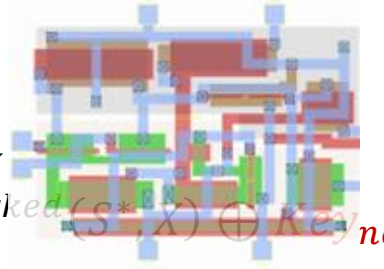
Easy for the designer, given the full knowledge of the FSM specifications

Execution Time

Key Evaluation

$Key_{next} = f(Key_{curr}, State), repeatedly$

Activation

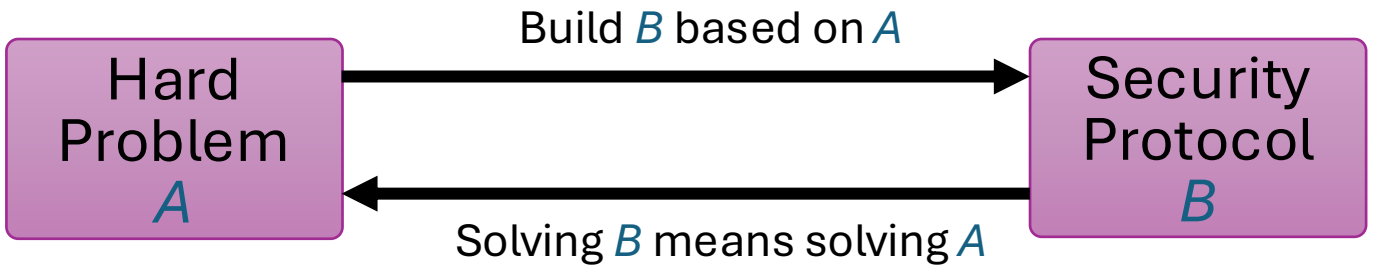


$Out(S^*, X) = Out^{locked}(S^*, X) \oplus Key_{next}$

Difficult for an attacker with only a netlist, as it requires multi-cycle functional reasoning.

Principles Behind White Box Defenses

- Moving the security foundation from empirical attacks to proofs
- Security reduction has played a key role in cryptography
  - It allows for cryptographic protocols to prove security requirements are met



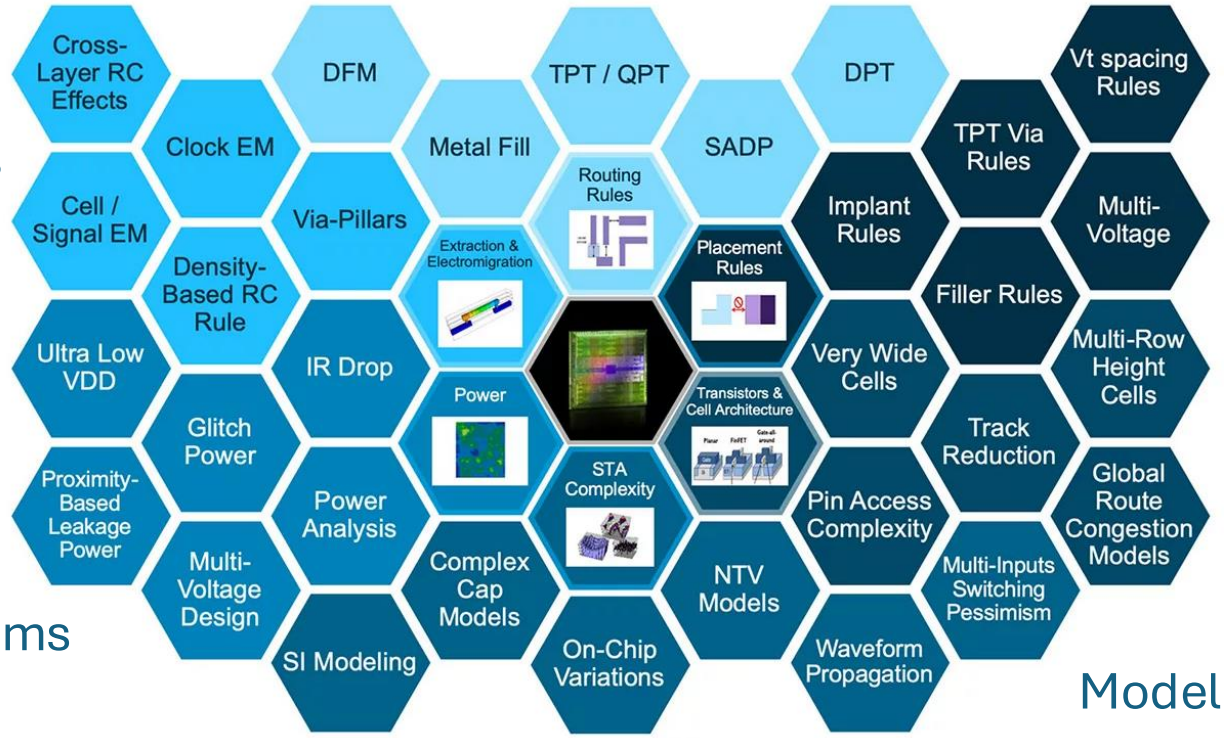
The *old problem A* is reducible to the *new problem B*

Candidates in the EDA area

- Many hard design and analysis problems exist in the EDA area
- Potential for some of them to be used in building security solutions

Manufacturing problems

Reliability problems



Design rules

Layout problems

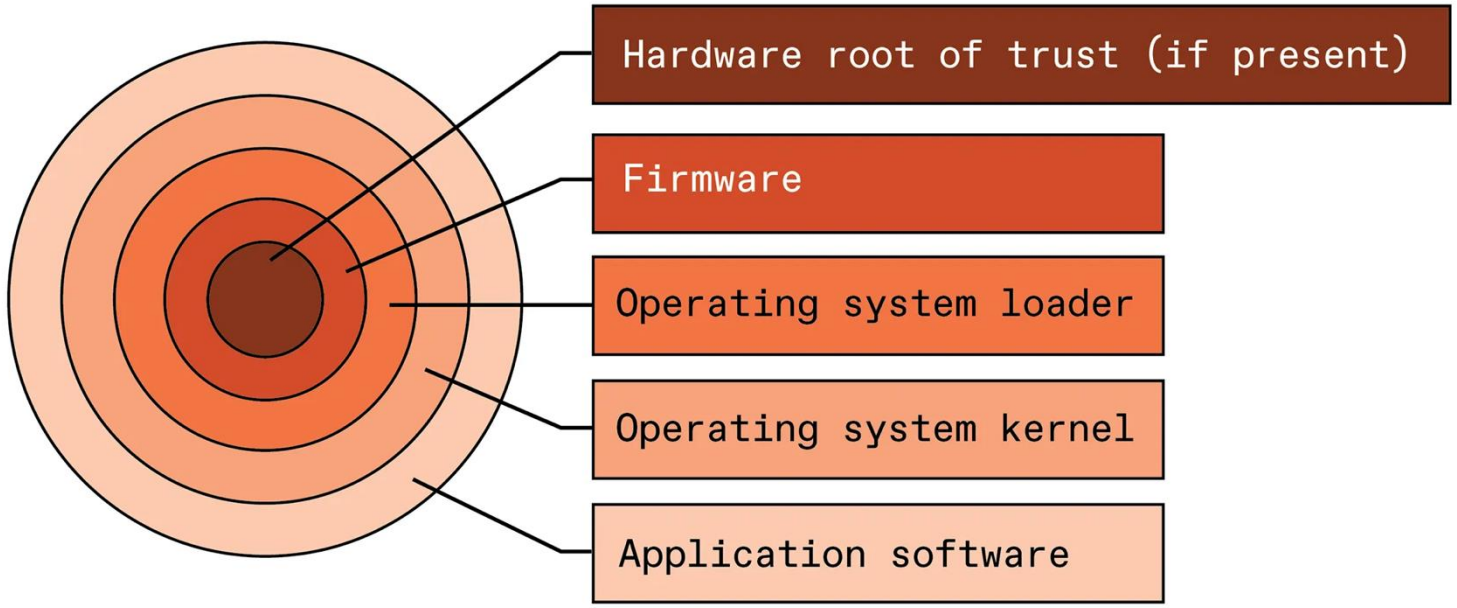
Power problems

Modeling problems

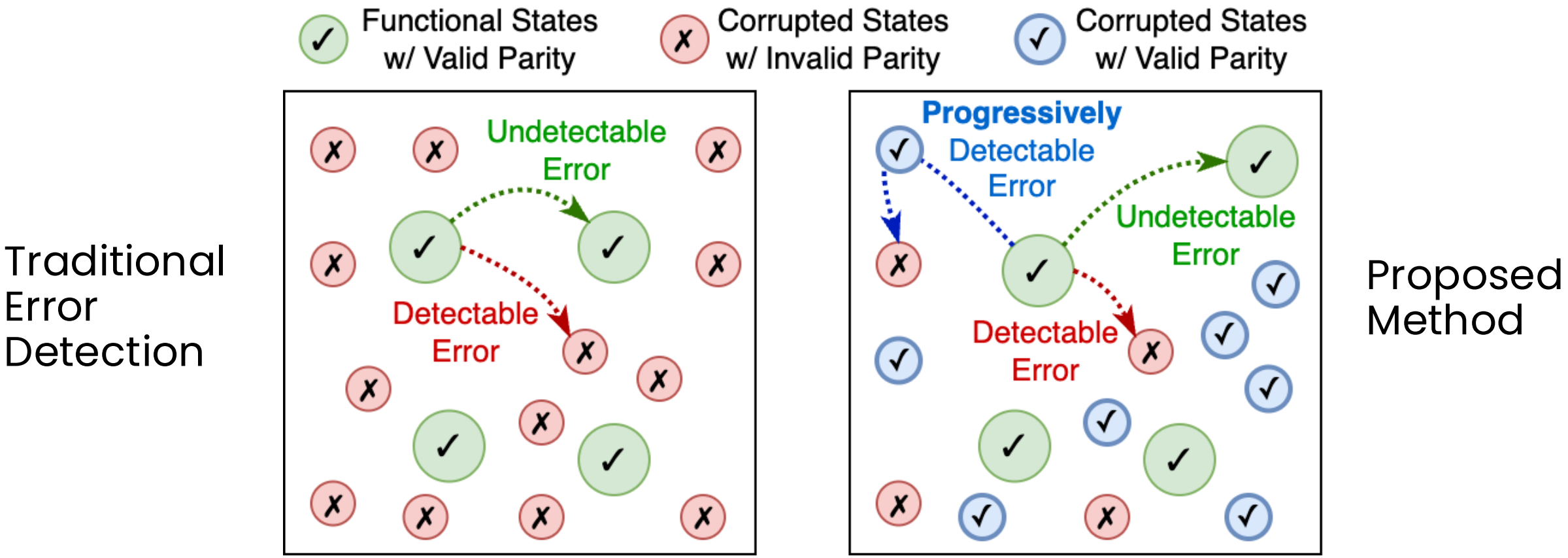
https://www.synopsys.com/implementation-and-signoff/resources/blogs/looking-past-

Defense-in-Depth

- A **Hardware Root of Trust (RoT)** is the foundational, immutable anchor of security embedded directly into the physical silicon of a device.
- It is the starting point for a system's chain of trust.
- Real-world security emerges not from one perfect defense, but from many imperfect defenses working together.



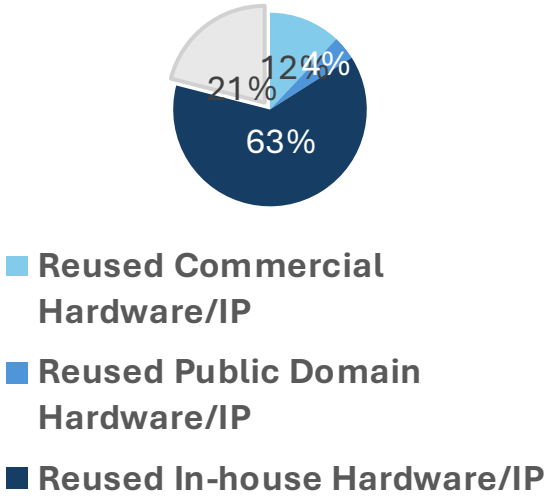
Bridging Security & Reliability



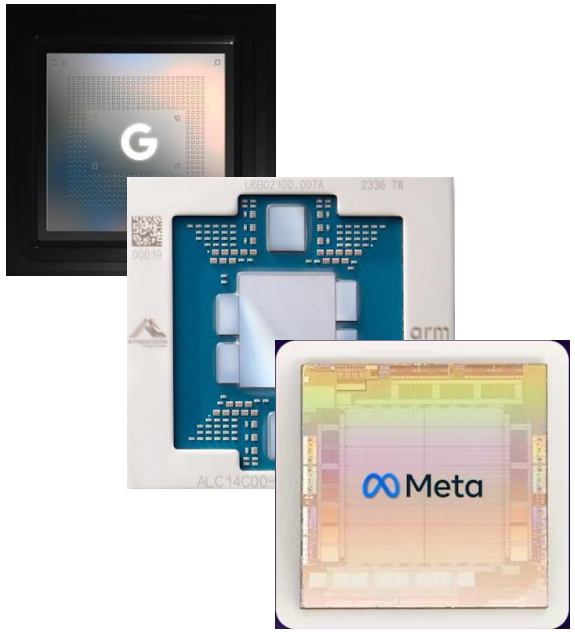
Breaking the one-shot detection barrier by allowing errors to propagate, creating multiple opportunities for detection over time.

Why Hardware Security Matters TODAY?

Growing Efficiency in Hardware Design



Custom Chip Design Embeds More IP in a Hardware Format



Pervasiveness of ICs in Society

Hardware security underpins the trustworthiness of modern society's critical computing infrastructure.

WASHINGTON STATE UNIVERSITY

A Primer on Hardware Security

Leon Li, EECS, Washington State University

5/19/2026