Permission to Use Conditions

1. Permission is granted to copy and distribute this slide set for educational purposes only, provided that the complete bibliographic citation and following credit line is included: “Hardware Security Slides by F. Gebali. ©2024. Gebali”.
2. Permission is granted to alter and distribute this material provided that the following credit line is included: “Adapted from Hardware Security Slides by F. Gebali. ©2024. Gebali"
3. This material may not be copied or distributed for commercial purposes without express written permission of the copyright holder.

ECE 448/548 Cyber-System Security

Hardware Trojans (HT)

F. Gebali

EOW 433

Office Phone: 250-721-6509

https://ece.engr.uvic.ca/~fayez/

Outline

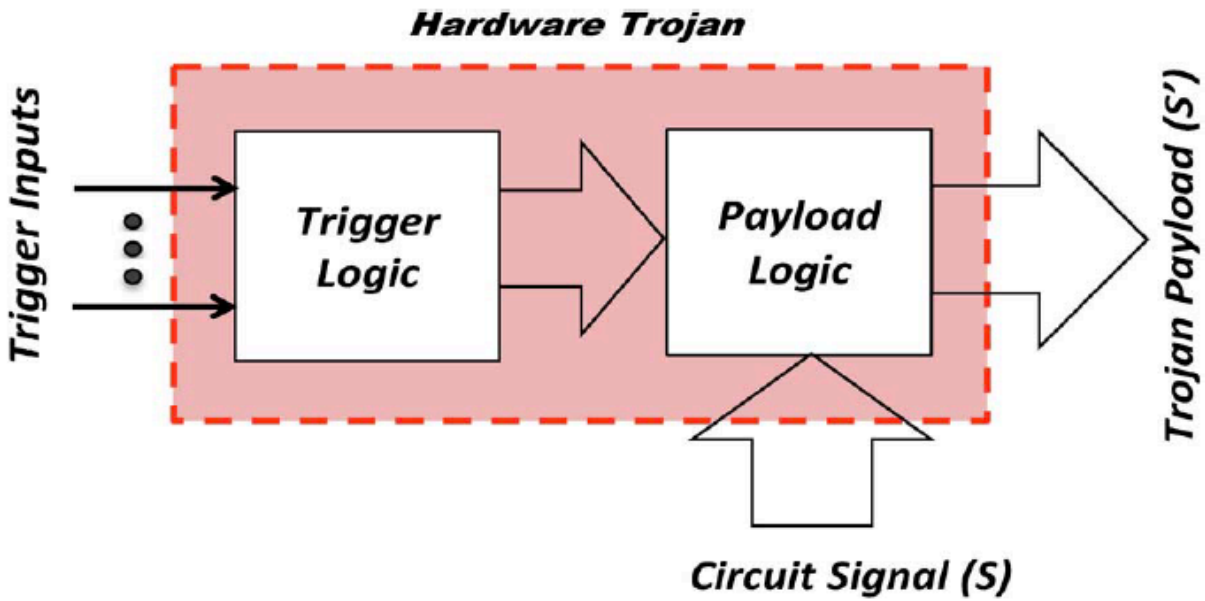
Introduction

Motivation

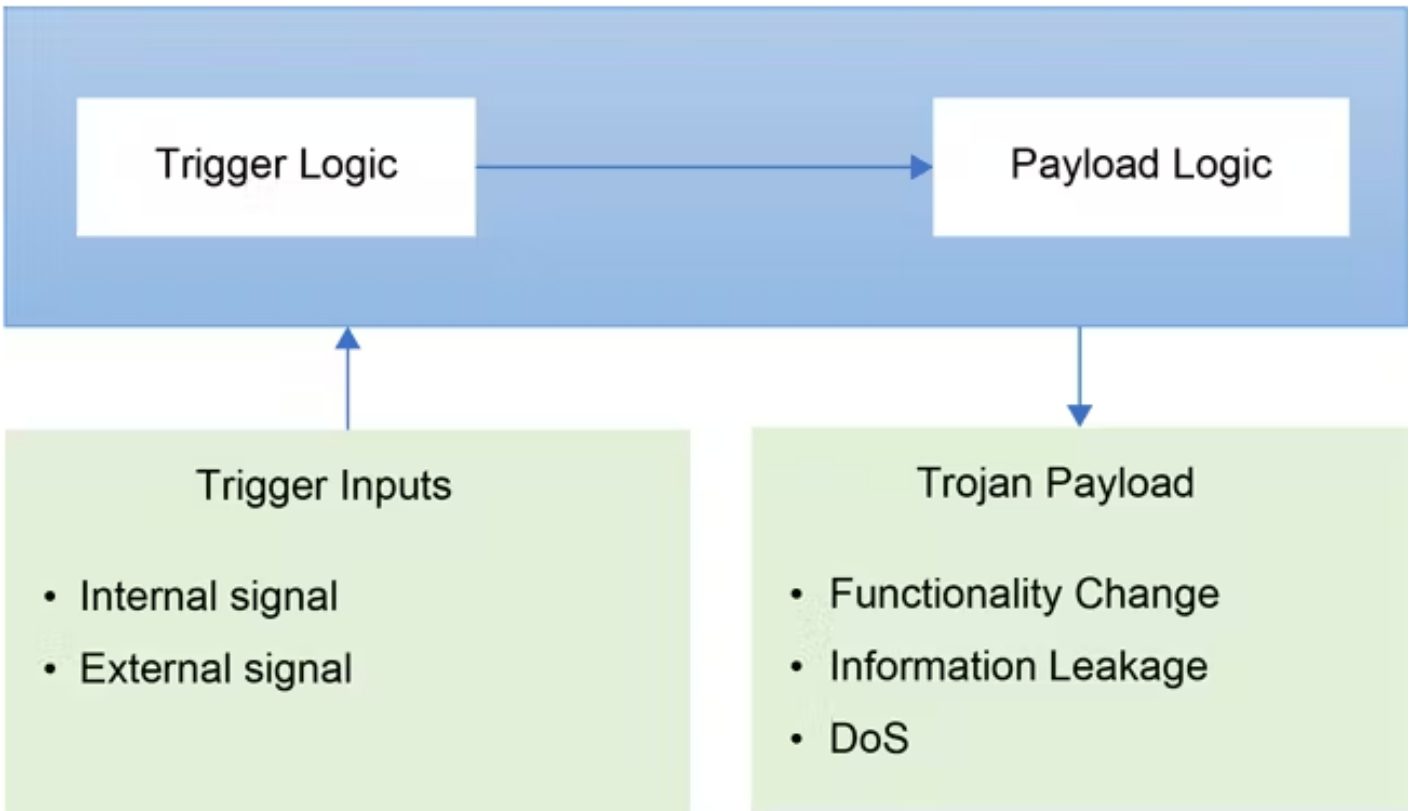
1. Horizontal business model is fabless design houses and few IC manufacturers
2. Traditionally computer system security is related to security of the software
3. Hardware was treated as being a root-of-trust (RoT)
4. Hardware trojans violated these assumptions
5. HT enable attacks without being detectable
6. Virus detection, pre-silicon validation/simulation or post-silicon test is incapable of revealing HT

HT Main Parts

1. Trigger
2. Payload



HT Operation



Comparing IC Faults vs. Hardware Trojans

	Fault	**Hardware Trojan**
Activation	Equivalent to line state (s-a-0 or s-a-1)	Combination/sequence of internal circuit states
Insertion Agent	Accidental due to imperfections in manufacturing process	Intentionally inserted during IC design or fabrication
Manifestation	Functional/parametric failure	Functional/parametric failure or information leakage

Comparing HT to Viruses

	Virus	HT
Location	RAM	Anywhere in system
Level	App level	Low-level
Detection	Virus scanners	HW/SW checkers, comparing with golden design/IC

Comparing Software Trojan vs. Hardware Trojans

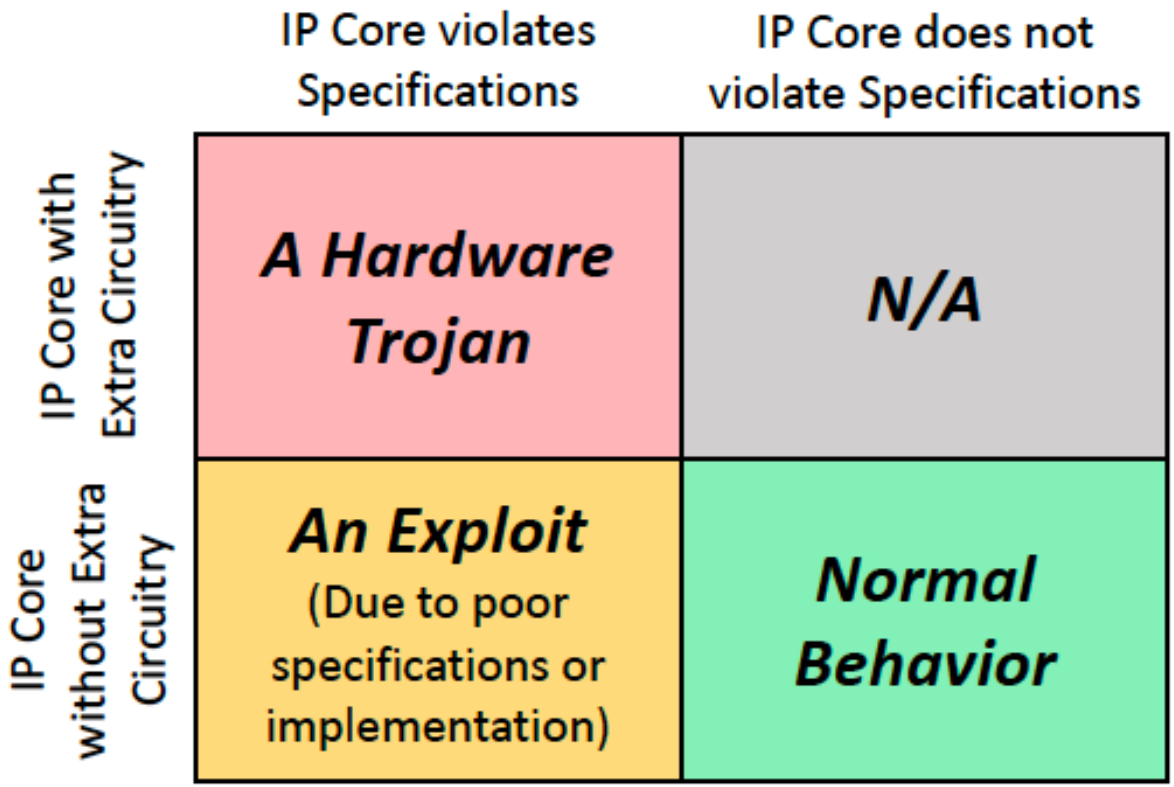
	Software Trojan	**Hardware Trojan**
Activation	A type of malware that resides in a code and activates during its execution and targets the OS	Resides in IC and activates during its operation
Infection	Acquired through user interaction e.g. downloading and running a file form the Internet	Inserted through untrusted entities in design or fabrication houses
Remedy	Can be removed in field through S/W support	Can not be removed once the IC is fabricated/configured

Hardware Trojans (HT)

Definition

Modifications in the hardware by adversary resulting in undesired behaviour.

IP Core Design Space [1]



Hardware Trojans

1. Attackers try to use the lowest abstraction layer to thwart detection
2. Hardware Trojan is a layer below the entire software stack
3. HT can bypass traditional defense mechanisms
4. Virus scan tools scan memory only and can not detect presence of HTs
5. HT attacks hardware and potentially software
6. HT could be single-purpose circuit or to enable high-level software control

Hardware Trojans Insertion Opportunities

1. Add small/midsize circuits at the HDL level.
2. Add gates at HDL level to create hidden side-channel to leak out secret keys [2]
3. Work below gate level by modifying silicon dopant level [3]
4. Work during manufacture at the layout level by modifying wire or dopants (DoS attack)

Goals of HT

1. Kill switch: hardware denial of service
2. Alter IC functionality
3. Leak sensitive information
4. Grant unauthorized remote control (e.g. privilege escalation [4])
5. Degrade performance

Malicious Modifications Examples

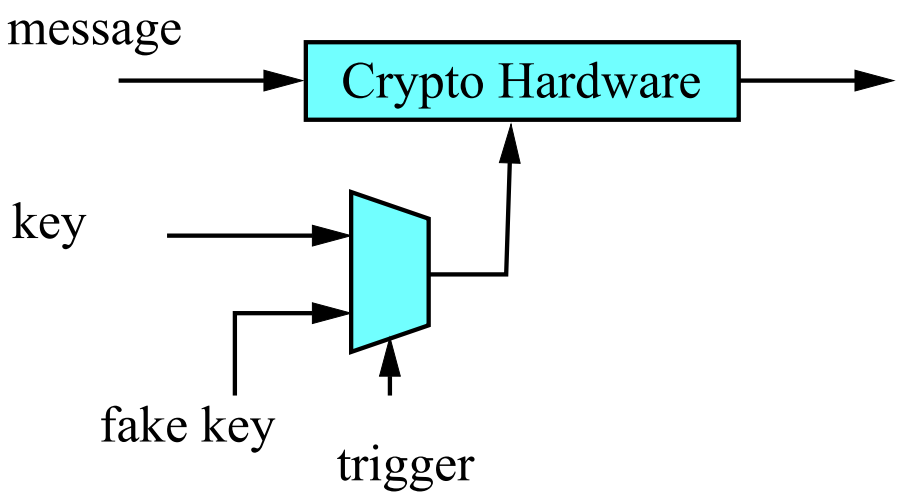
1. Intel has "Intel Management Unit" (ME) that can take full control of Intel computers but user can not disable it
2. JTAG controller with backdoor in Actel/Microsemi ProASIC3 A3P250
3. Seagate shipped external drives that can steal data
4. European processor with a remote kill switch
5. Counterfeit Cisco routers in US defence & finance

Trojan Example: General-Purpose Processor [5]

1. Change instruction execution order
2. Override memory range protection
3. Change PROM content (e.g. BIOS)
4. Build frontdoor to help a software adversary
5. Intercept and modify I/O data
6. Freeze or change timing or skew of the clock grid

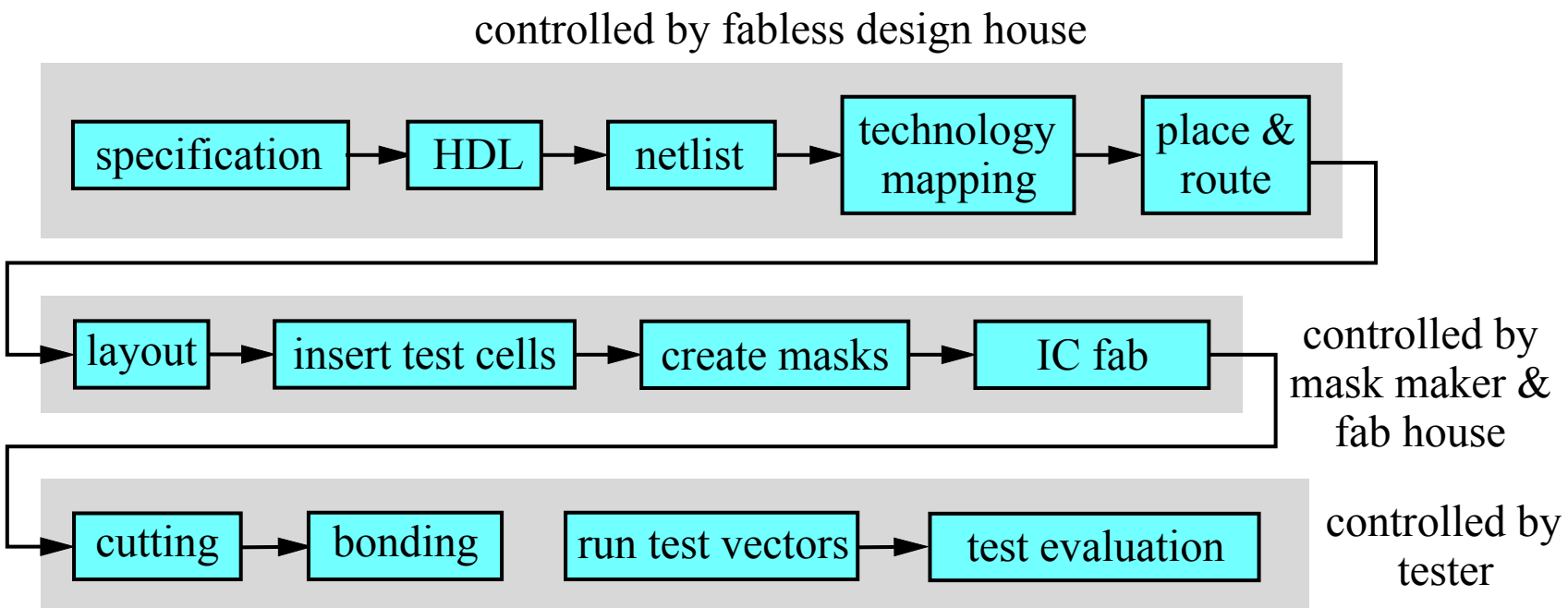
Trojan Example: Crypto Processor [4]

1. Do what is done to a GP processor
2. HT could issue predefined dummy keys instead of the randomly generated keys
3. Leak secret information by modulating secret key and send signal over power port
   and lower power when transmitting a 0

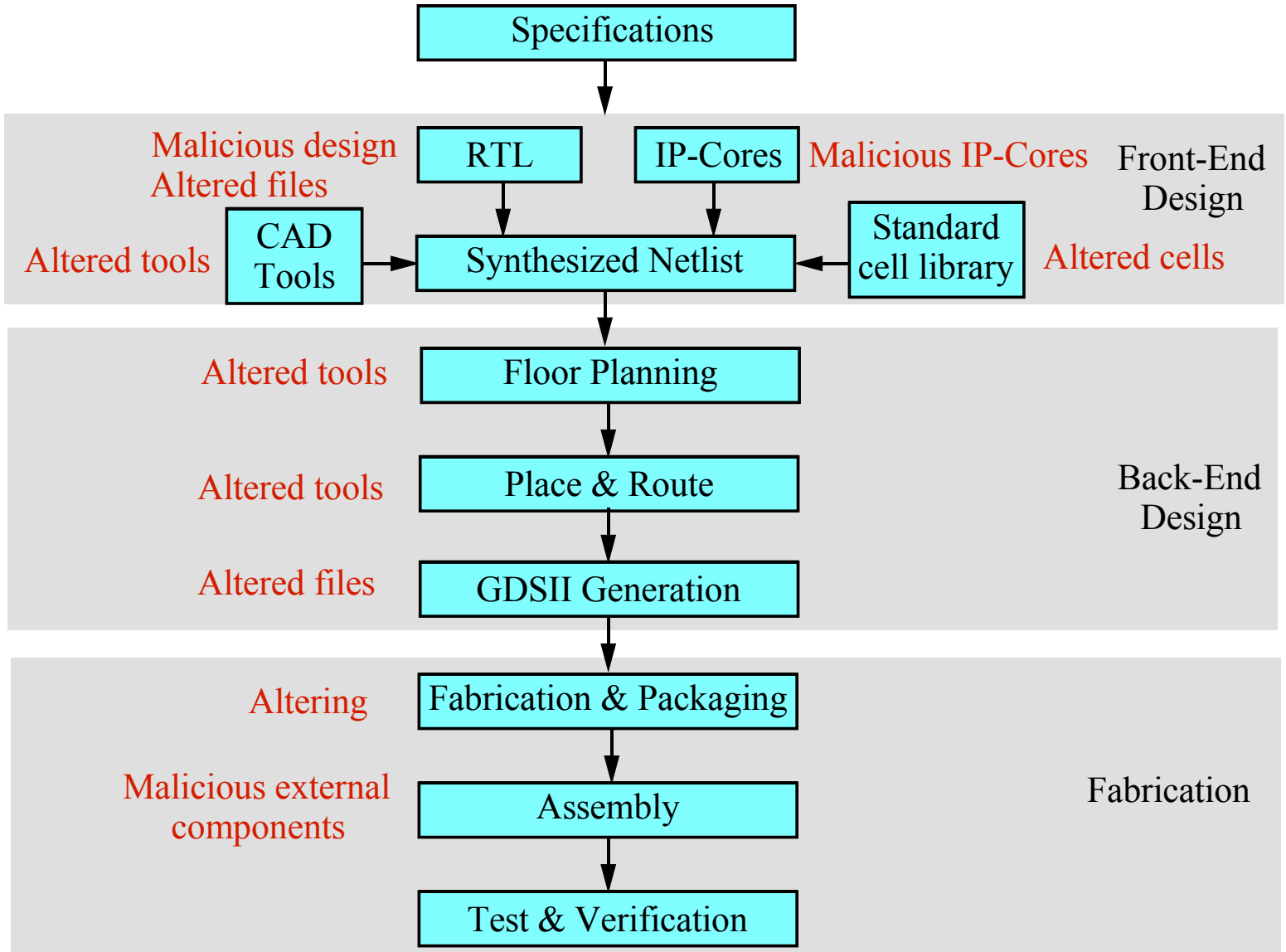


Insertion of HT

IC Design Flow [7]

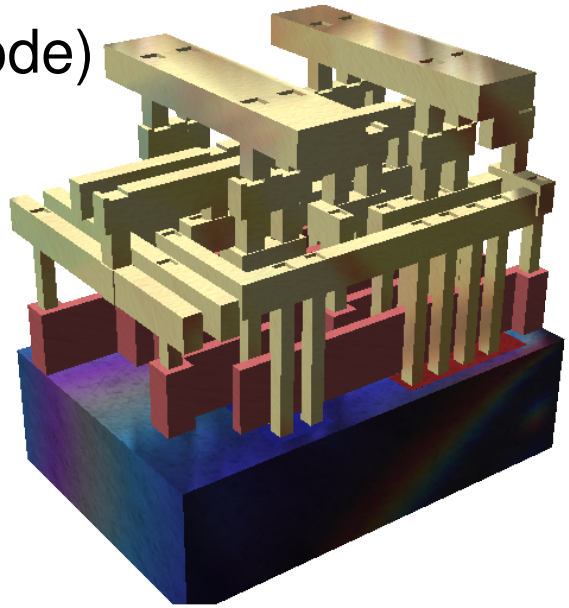


IC Design Flow [7]



Seven HT InsertionOpportunities

1. Design house teams (HDL source code)
2. Third-party IP (3PIP) (JTAG, HDL source code)
3. Untrusted CAD tool vendors
4. Untrusted fab house (GDSII files, doping)
5. Testing stage attacks
6. Distribution stage attacks
7. FPGA chips (modify configuration file)



1. Design Team

1. Motivation: Steal info, DoS, facilitate future attacks
2. Resources: HDL design files
3. Feasibility: Easy to implement HT
4. Detectability: Code checkers

2. 3PIP Vendor

1. Motivation: Less security, steal info, DoS, future attacks
2. Resources: IP design files and source code
3. Feasibility: Easy to insert HT, limited control over triggering
4. Detectability: Formal verification & code analysis

3PIP types:

1. Soft (HDL/RTL)
2. Firm (netlist level)
3. Hard (GDSII/layout level)

3. CAD Tool Vendor

1. Motivation: Backdoor entry, time bomb, steal info
2. Resources: Design files & source codes
3. Feasibility: Direct source code modification
4. Detectability: Side-channel analysis

4. Foundry

1. Motivation: Lower reliability, steal info, DoS
2. Resources: GDSII layout (masks)
3. Feasibility: Difficult to figure function or modify masks
4. Detectability: Side-channel analysis, mask inspection

5. IC Test

1. Motivation: Hide HT detection results
2. Resources: Collected test results, functional specs
3. Feasibility: Modify test results
4. Detectability: Difficult to detect

6. IC Distribtor

1. Motivation: Replace IC with HT-infected IC
2. Resources: Functional specs
3. Feasibility: Difficult to add HT to IC (not silicon)
4. Detectability: Using PUFs

7. FPGA Chips

1. Motivation: DoS, steal info
2. Resources: Design files, intercept reconfiguration updates
3. Feasibility: During design or during remote updates
4. Detectability: Side-channel analysis

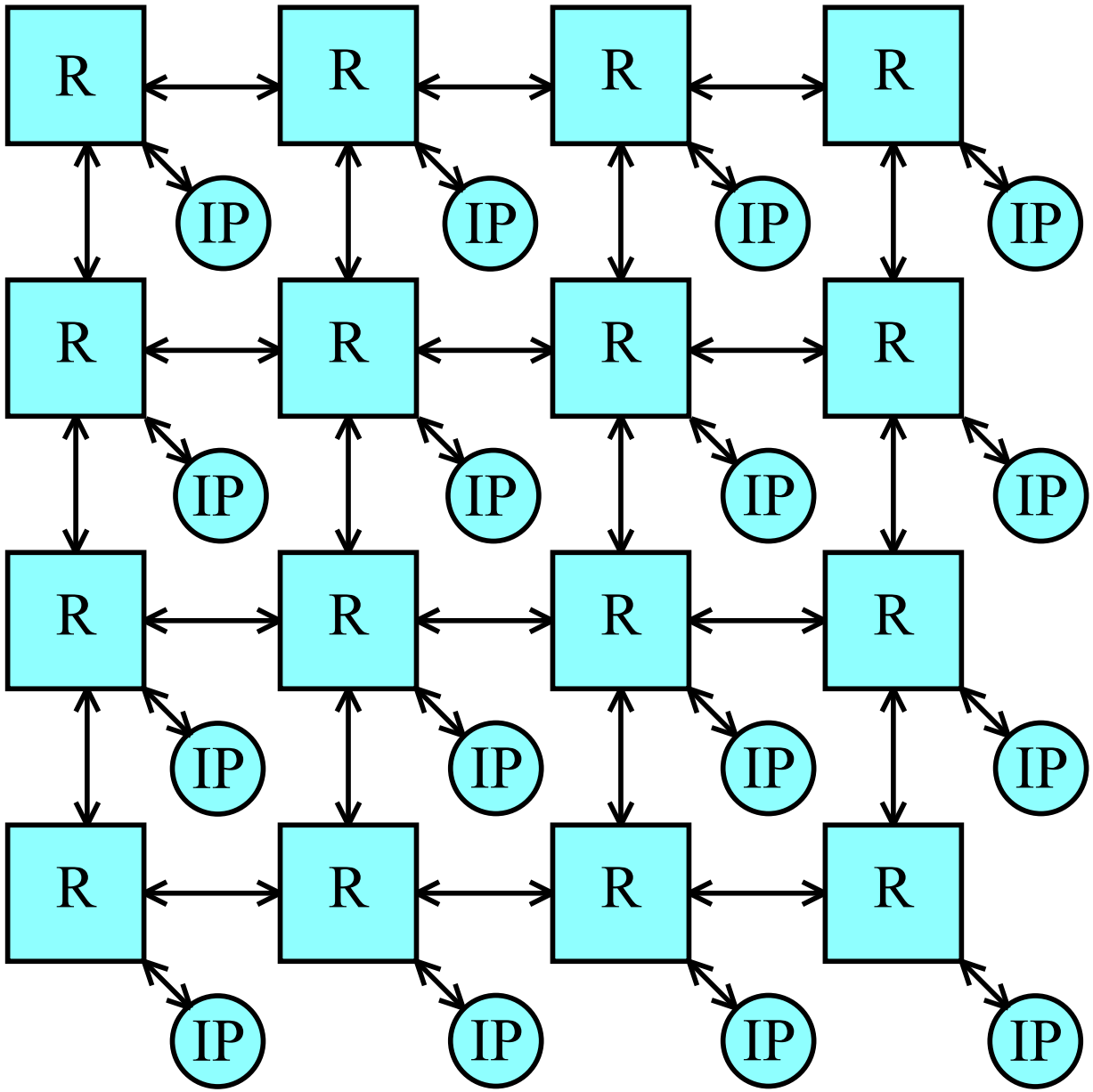
HT Location [5]

1. ALU
2. Controller
3. Memory
4. I/O
5. Power supply
6. Clock tree
7. IC fabrication parameters

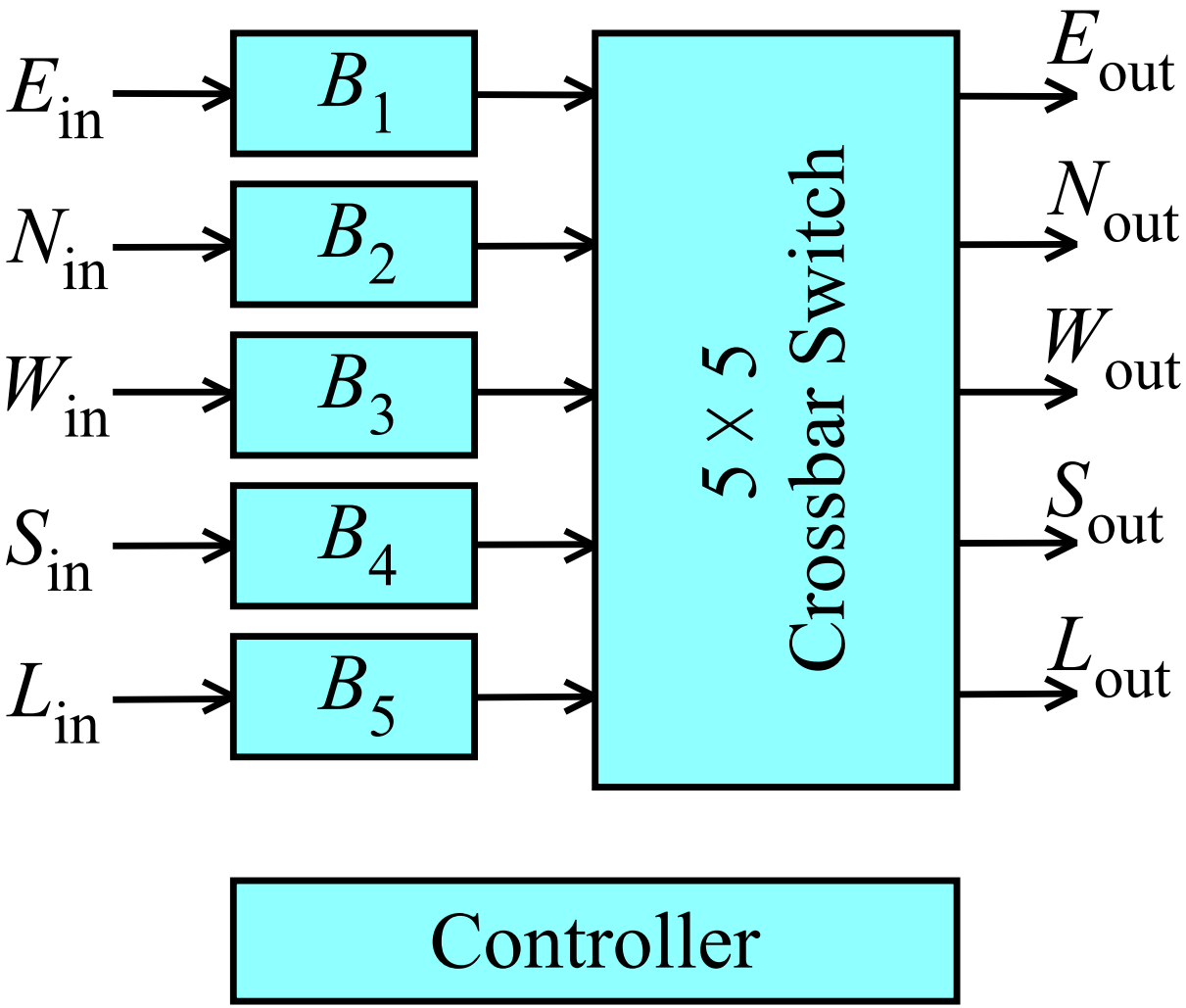
What can be Altered

1. Specification
2. HDL source code
3. Netlist
4. Chip timing
5. IC layout
6. IC doping

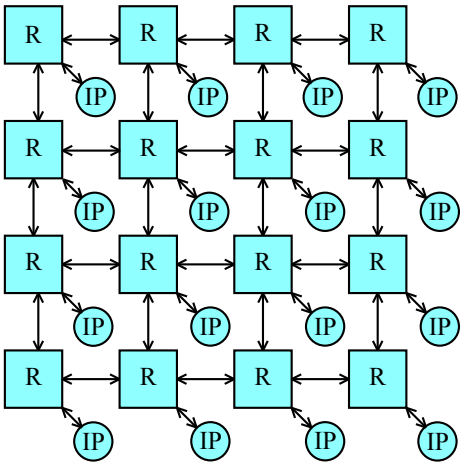
System on Chip (SoC) Example: Mesh NoC

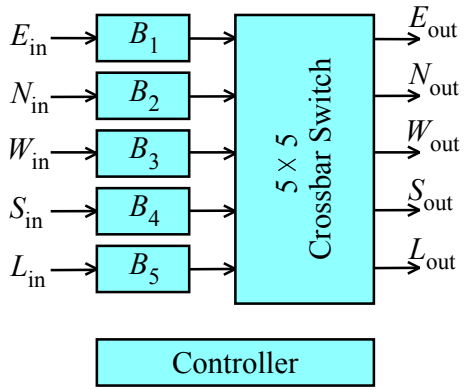


Network on Chip (NoC) Router Example: Input Buffer



Normal Router Actions when Trojan-Free



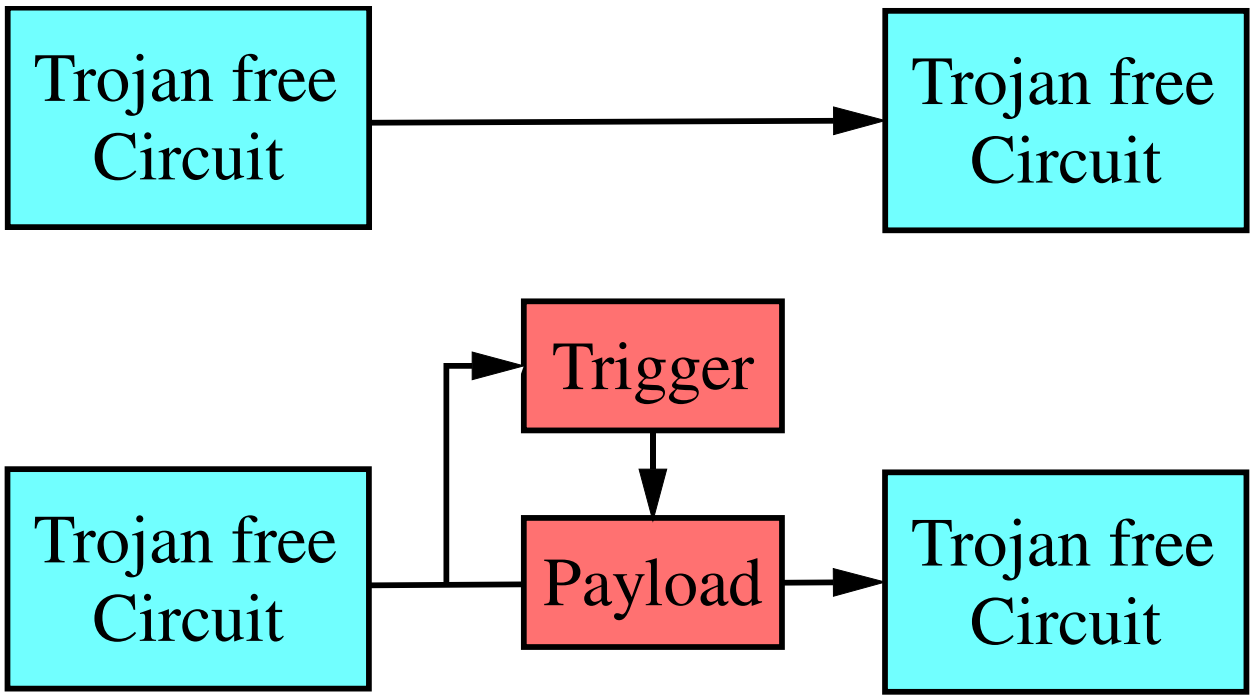


1. Choose head-of-line (HoL) packet from one of B_1 to B_5
2. Read header and determine output link (E, N, W, S, L)
3. Issue REQ and wait for ACK from next router
4. Configure switch matrix (connect buffer output to outgoing link)
5. Transmit HoL and update buffers on both switches

Trojans Targeting the NoC Router

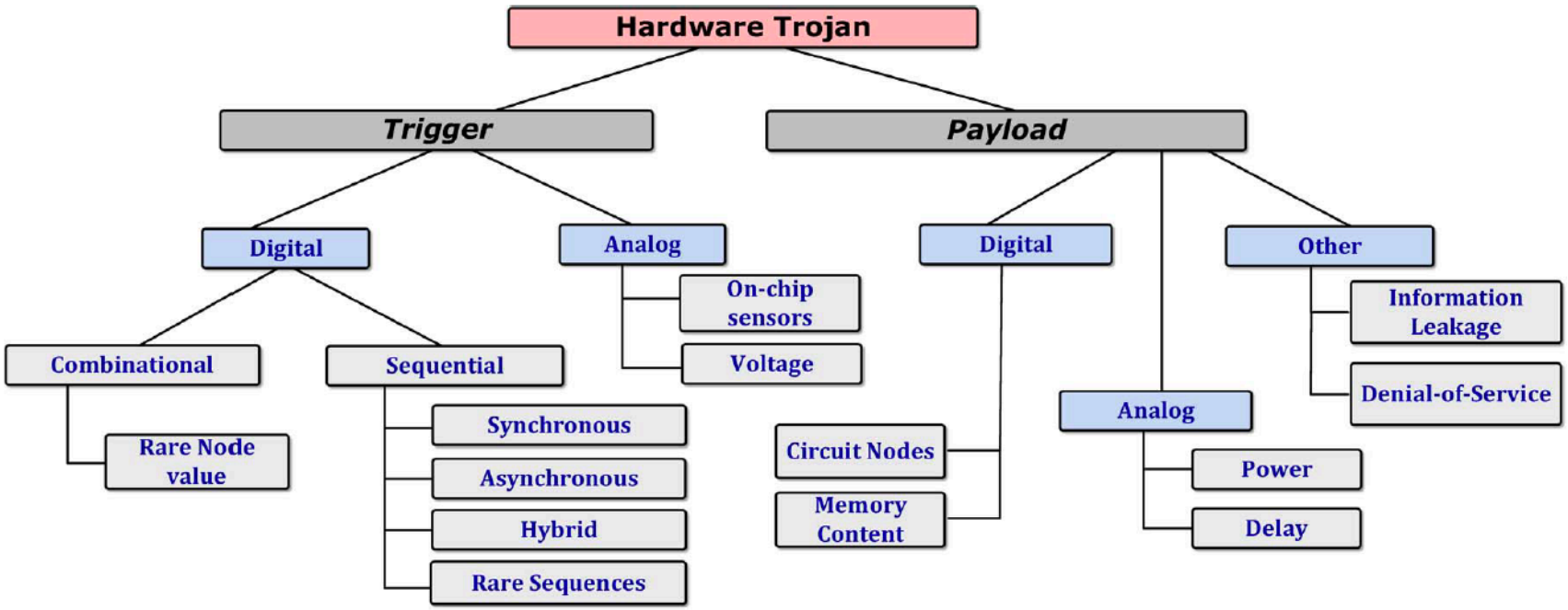
1. Target the packet payload or header to modify
2. Deadlock (packet waits for ACK that never arrives)
3. Live lock (packet never reaches destination, circulating addressing strategy in two or more switches)
4. Information leakage (copy payload)
5. Replay (extra requests, flood network)
6. Misrouting including blackhole attack

Hardware Trojan Structure



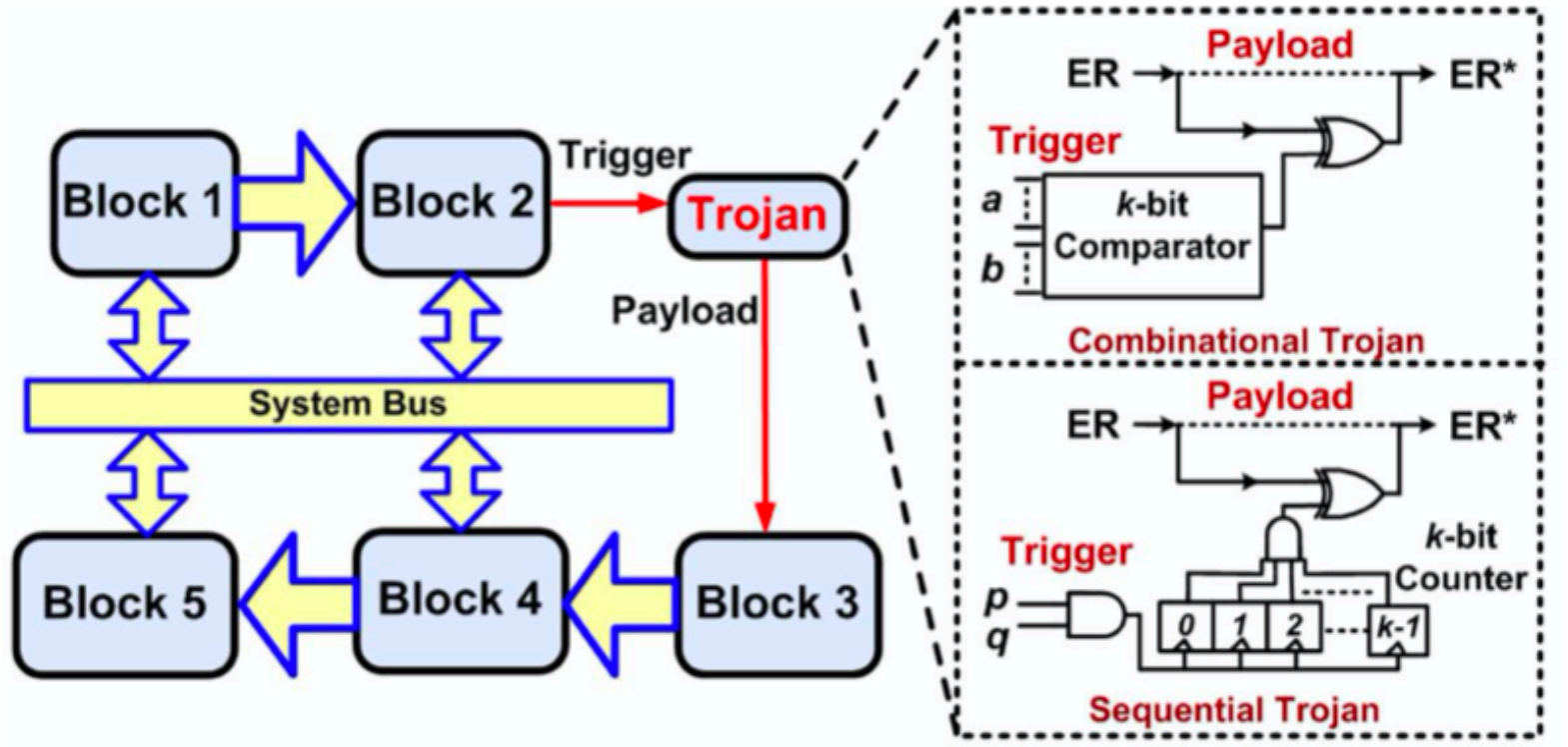
Trigger is a rare event and could be combinational or sequential

Trojan Taxonomy [6]



Triggering

Trojan Trigger : Combinational vs. Sequential



Triggering [8]

1. Always on
2. Based on combinational logic
3. Based on Sequential logic
4. Count certain event
5. After a time period
6. At random
7. Remote command

Triggering

1. Designed to activate under rarely occurring events (Follows Geometric Distribution)
2. Time-based internally triggered
3. Physical condition internally triggered
4. User input externally triggered
5. System output externally triggered

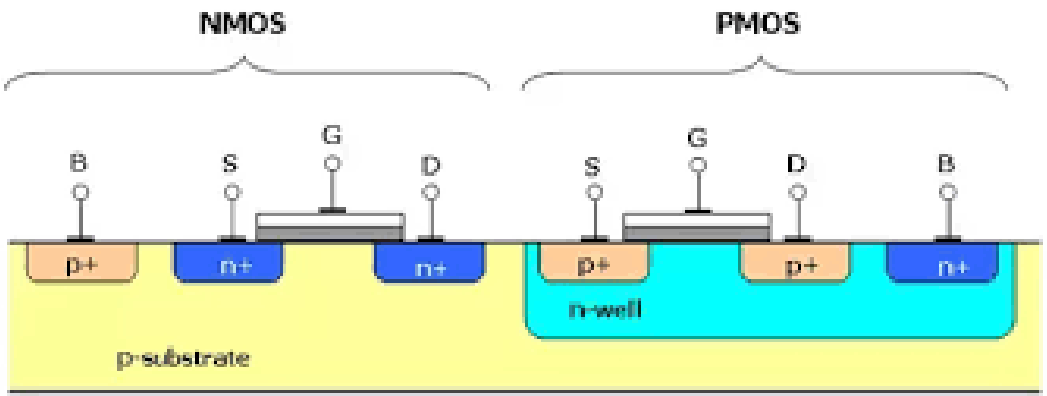
Trojan Payload

Trojan Payload [8]

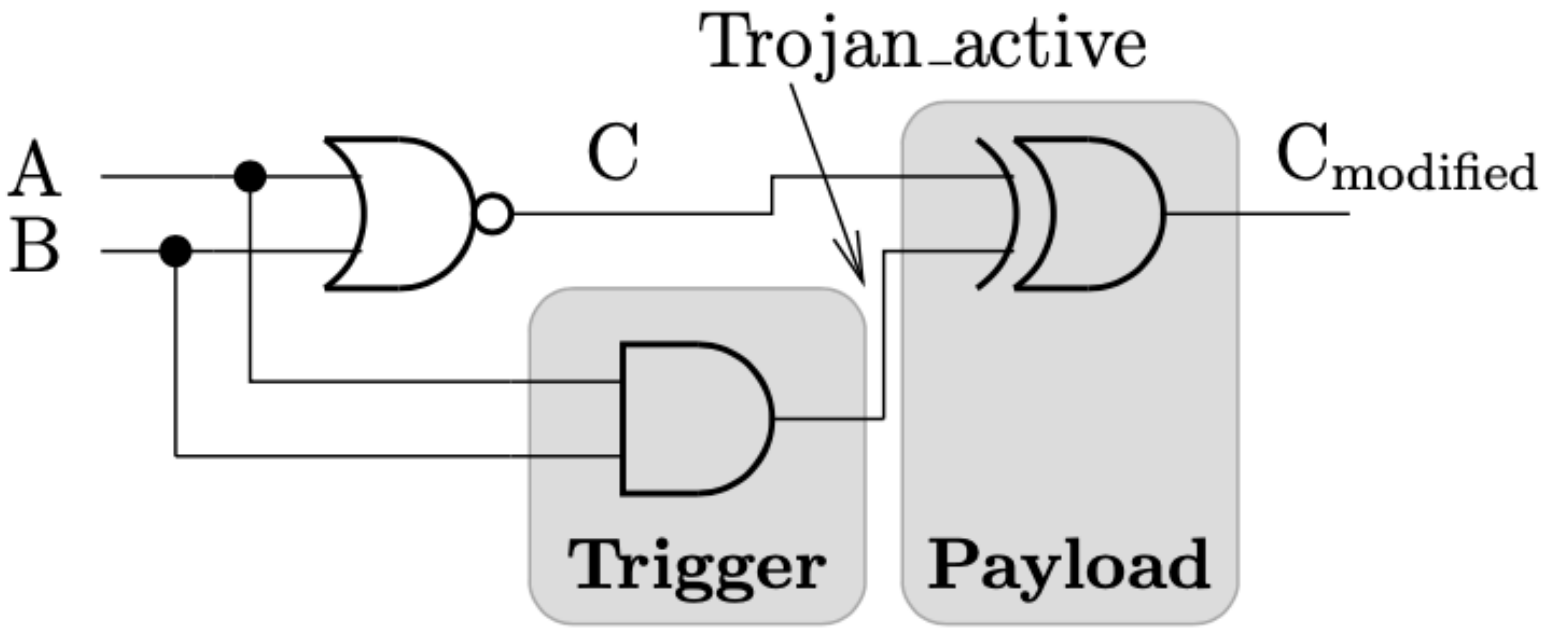
1. Change functionality
2. Degrade performance
3. Leak information
4. Deny service
5. Support design of software-based attacks
6. Bypass memory management unit
7. Shadow mode: login backdoor or steal passwords

Trojan Payload: Reduce Reliability

1. Reduce the width/height of a metal line to increase the rate of electromigration Impossible to measure width of the wire as drawn on a faulty mask
2. High spike at MOS gate to cause gate oxide short
3. Change doping of substrate or other layers



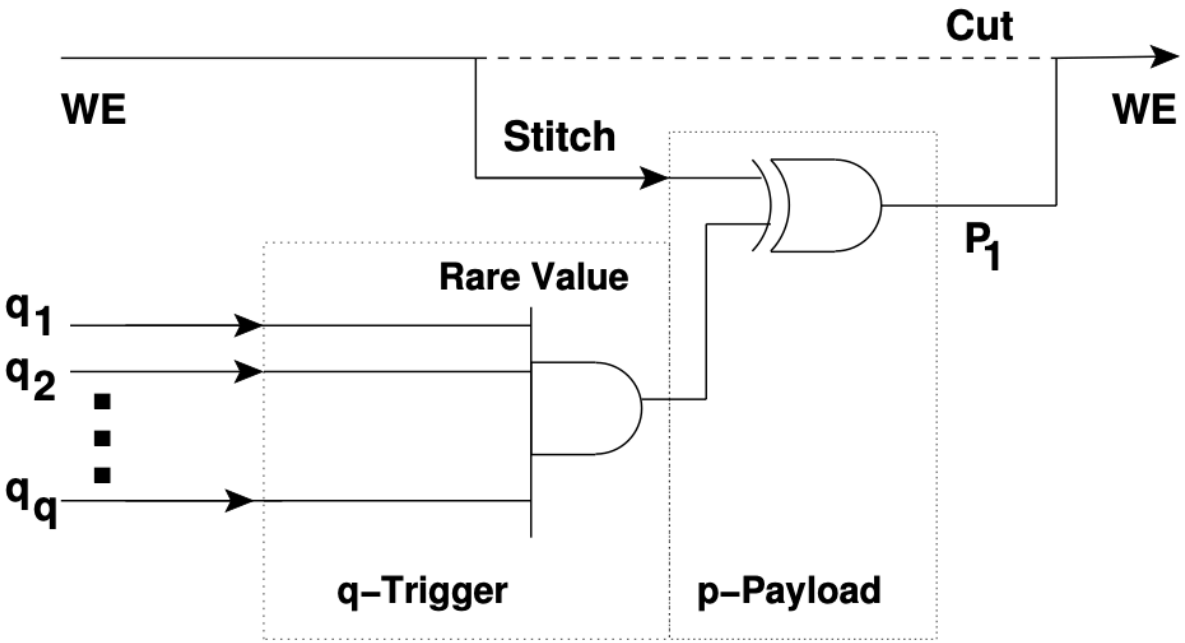
Trojan Payload: Randomly Change Signal Value



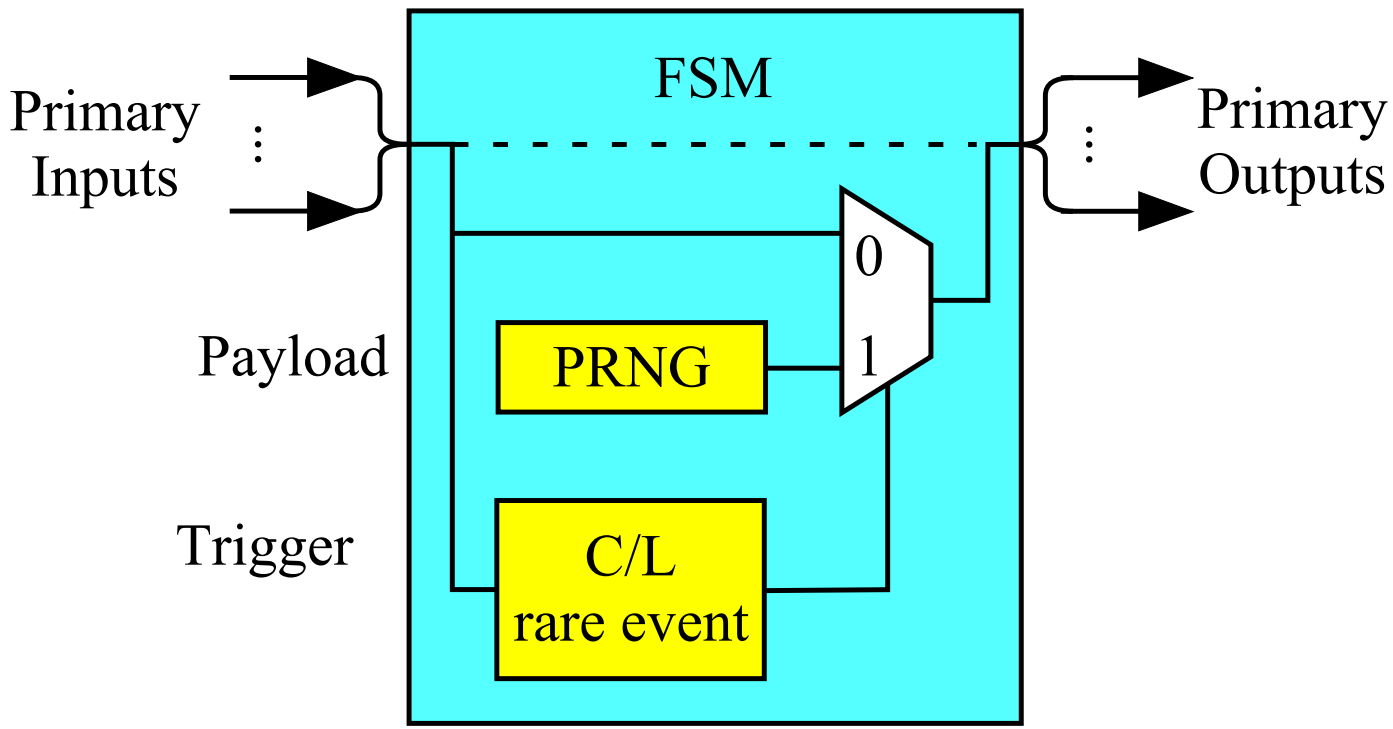
Trojan Payload: Affect Memory[9]

1. Modify memory protection through MMU
2. Modify data or address bus (man-in-the-middle)
3. Modify data bus for write operation
4. Modify data bus for read operation
5. Modify address bus

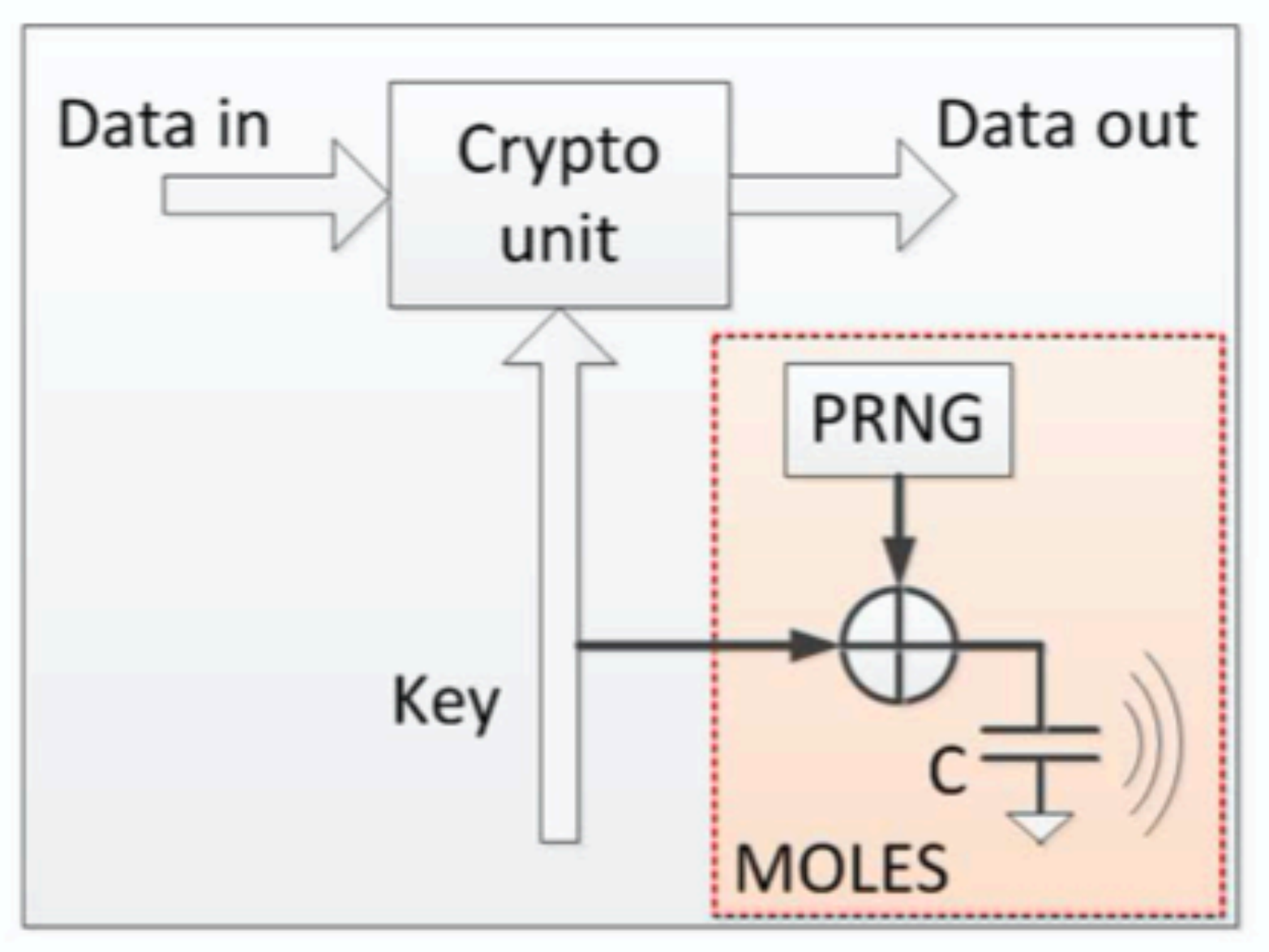
Trojan Payload: Erratic Behaviour of Memory



Trojan Payload: Erratic Behaviour of FSM



Trojan Payload: Facilitate Side-Channel Attack



Detection

Challenges of Trojan Detection

1. What type of Trojan? (Trojan model)
2. Activating the Trojan (test generation)
3. Eliminating noise, RPV, for side-channel analysis (SCA)

Trojan Detection Techniques: Logic Testing

1. Formal verification/code checkers: check specifications
2. Side-channel analysis (SCA): Delay, power, radiation
3. Structural testing
4. Use statistical test pattern generation
5. Online Assertion chekers

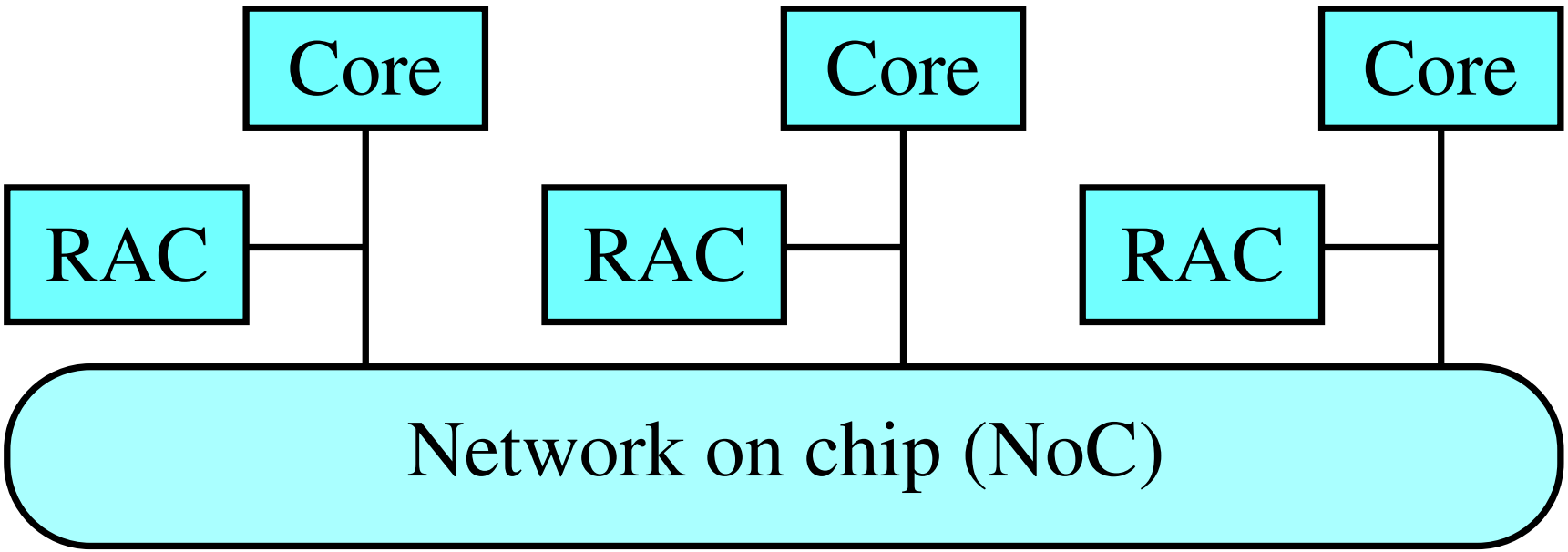
Formal Verification

1. Enables highest security level in the common criteria framework
2. Formal verification uses temporal logic to deal with time-based specifications
3. Can be used to check correctness of protocols, combinational circuits, sequential circuits and source code
4. Software tools include property specification language (PSL), SystemVerilog Assertions (SVA), SystemC, AVISPA, etc
5. Checks V & V:
    1. Validation: Design satisfies user's needs
    2. Verification: Design satisfies spefications

Invasive HT Detection Using Formal Verification

1. Design specifications are expressed as properties
2. Corner cases could be expressed as properties too
3. Add extra logic to monitor delay times [18]
4. Add programmable assertion checkers (PAC) to monitor operation [12]

HT Detection Using Formal Assertion Checkers



RAC: reconfigurable assertion checker

Pre-Silicon Trojan Detection Using Code Coverage Analysis

1. Code coverage is percentage of executed code lines during functional verification
2. Can be used to identify suspicious signals and gates
3. Code coverage is also used to isolate code lines not used during functional verification

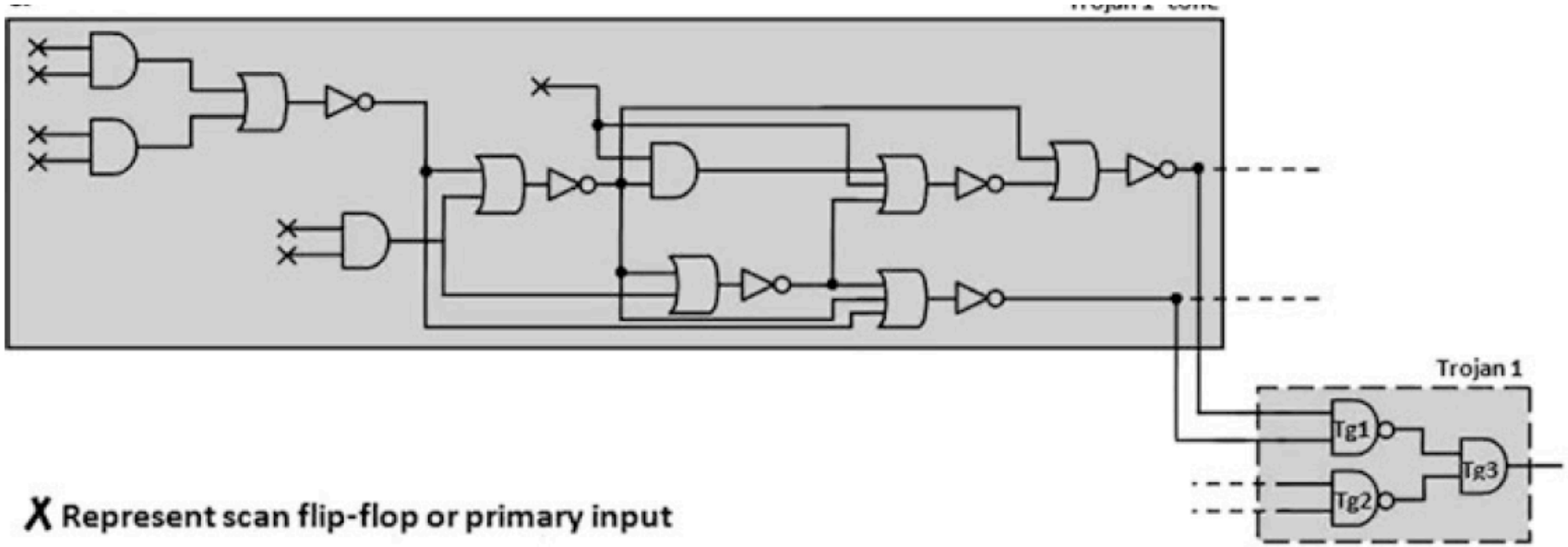
Pre-Silicon Trojan Detection Using Functional Analysis

1. Apply random patterns and activate Trojan
2. On other hand, logic testing applies specific patterns not designed to activate Trojans
3. Functional analysis could find nets that rarely switched

Limitations of Functional Analysis

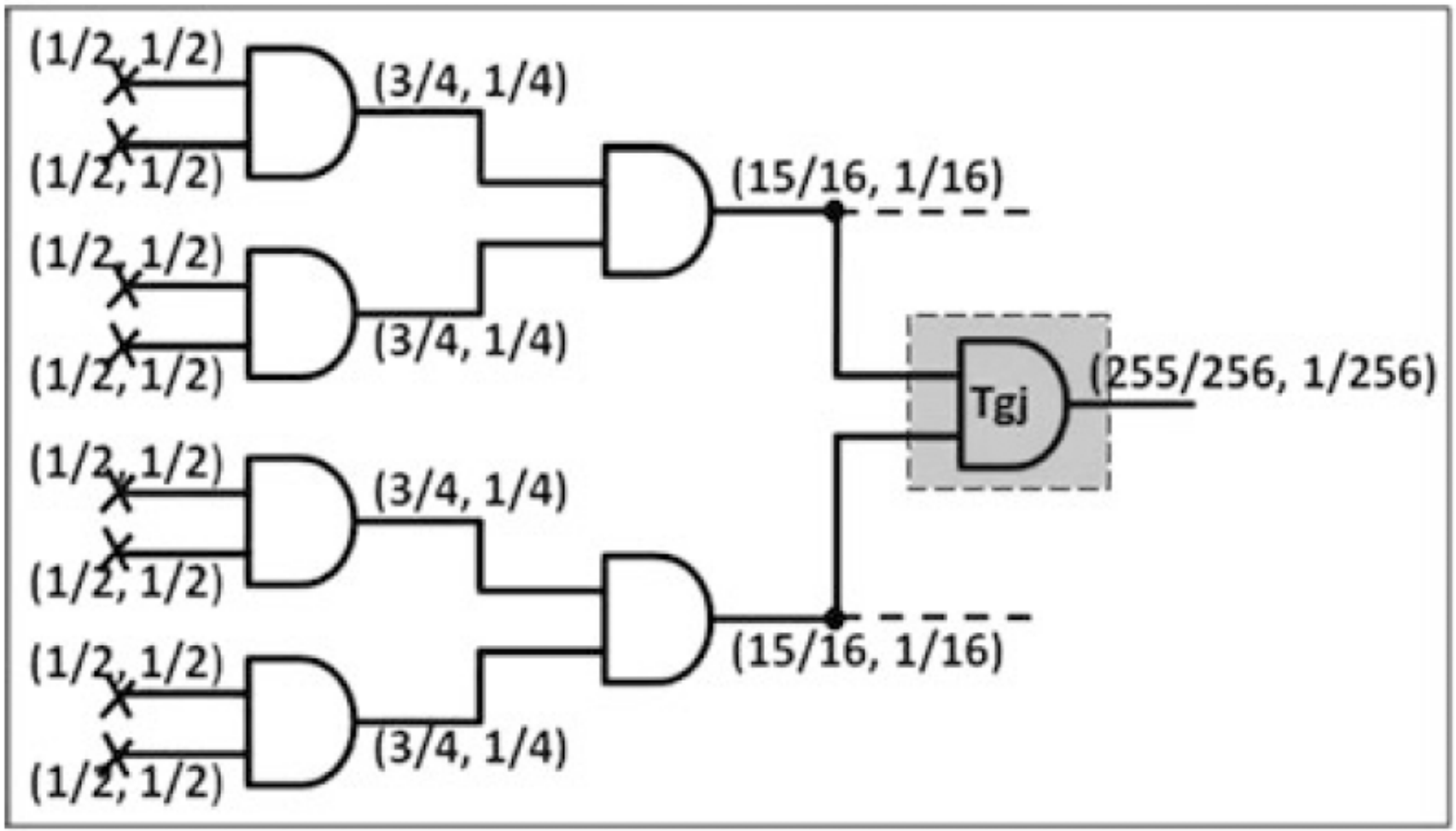
1. Automatic test pattern generation ATPG are designed to detect logical defects based on a given netlist
2. ATPG is not designed to directly detect HT activation or detection
3. HT are designed to be inactive most of the time and circuit appears to operate normally
4. HT target circuits with low controllability or observability

Defining HT Cone [17]

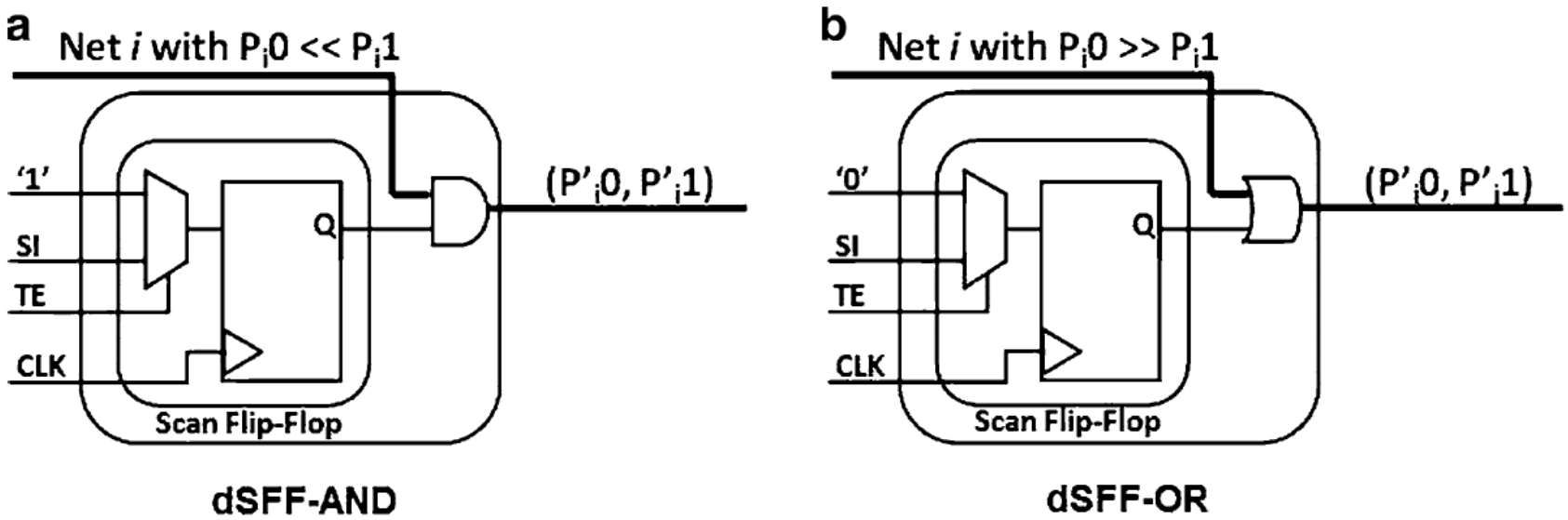


1. Cone has 17 gates in 11 levels
2. After 1,000 random inputs, 67 transitions at Tg1 input.

HT Transition Probability: Low Trigger Activation Probability



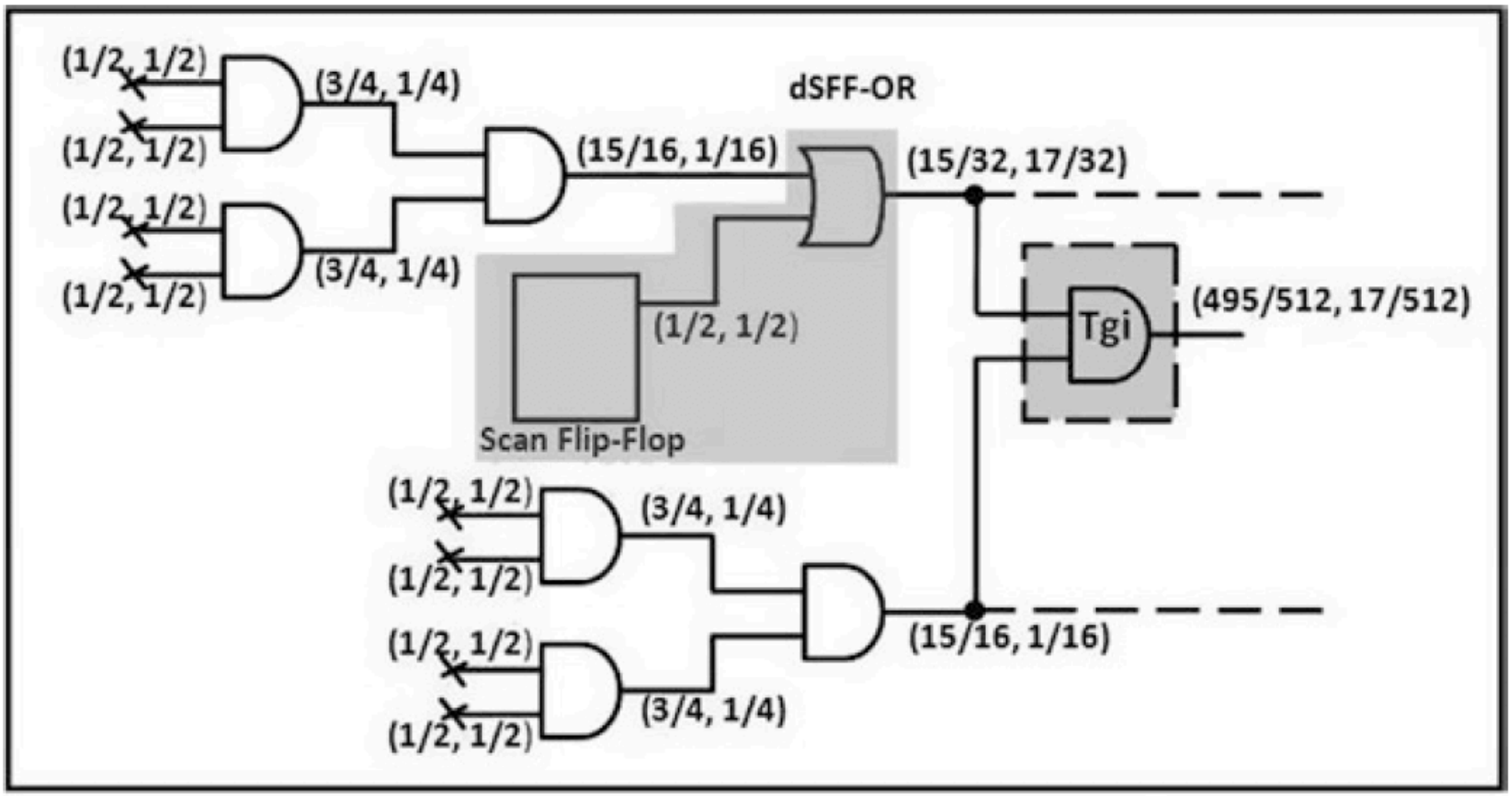
HT Transition Probability: Dummy Flip-Flop



Assuming at inputs $p_0 = p_1 = 0.5$:

1. dSFF: dummy scan flip-flop
2. For AND gate $p_0 = 0.75$ and $p_1 = 0.25$
3. For OR gate $p_0 = 0.25$ and $p_1 = 0.75$

HT Transition Probability: Dummy Flip-Flop



Estimated time to Activate Trojan

1. Assume output of Trojan cone has probabilities p_0 & p_1
2. Average number of times when trigger is active is

$$\begin{aligned} n_a &= \sum_{i=0}^{\infty} i\, p_0^i\, p_1 \\ &= \frac{p_0}{p_1} \end{aligned}$$

3. Trojan designer aims to use rare event situation where

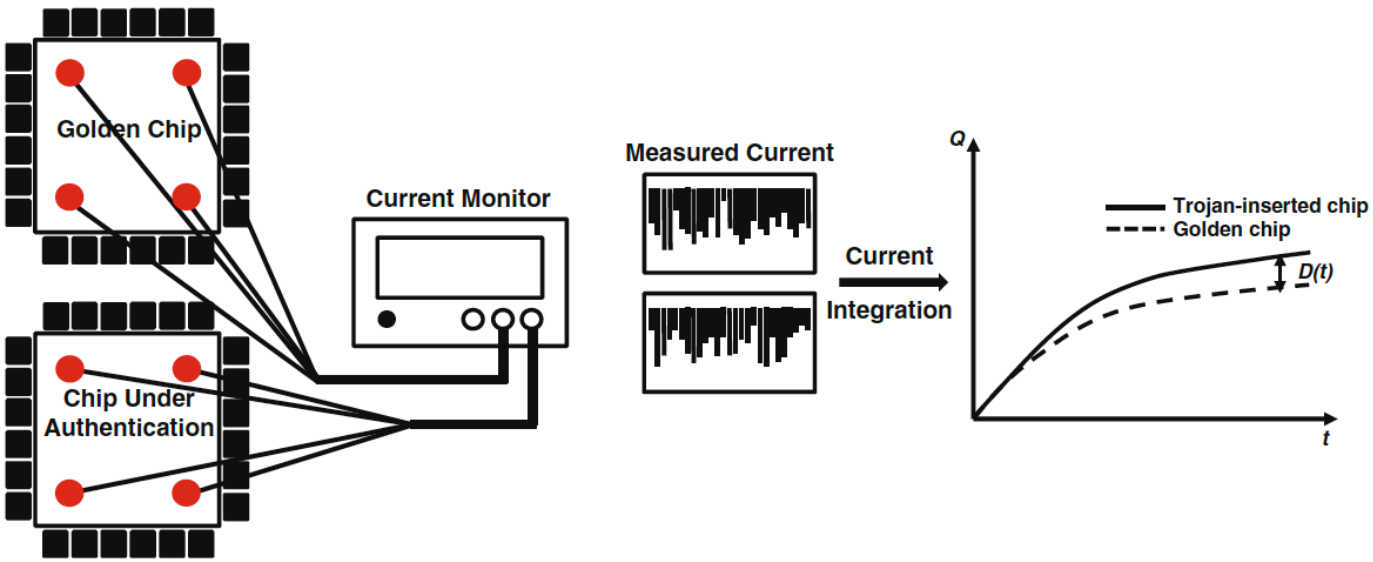
$$p_0 \to 1 \qquad \text{and} \qquad p_1 \to 0 \qquad \text{yielding} \qquad n_a \to \infty$$

Trojan Detection Techniques: Side-Channel Analysis

1. Measures supply current or path delay
2. Relies on large parameter variations
3. Require golden design or ICs for comparison
4. Signal-to-noise and trojan-to-circuit ratios. This can be increased by observing parts of the IC

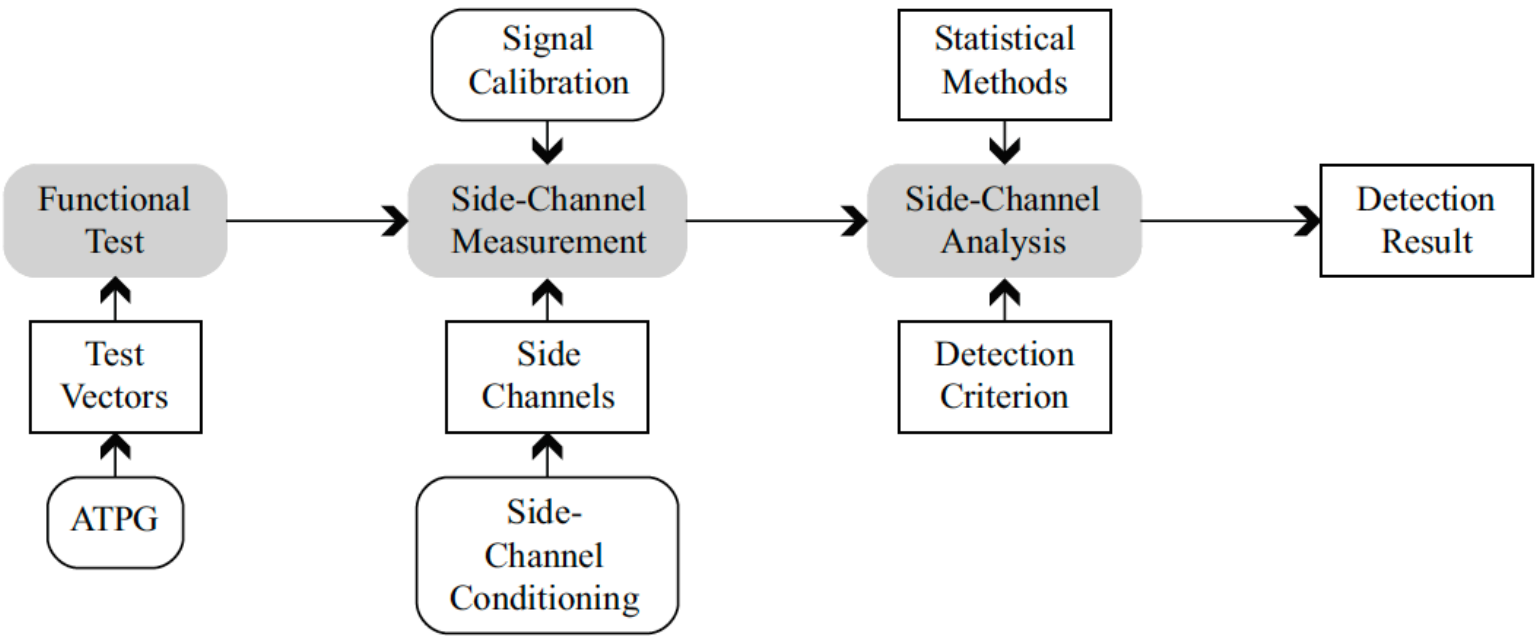
Trojan Detection: Side Channel Current Integration

1. Assumption is Trojan circuit increases current drain
2. Assumption is Golden Trojan-free circuit is available

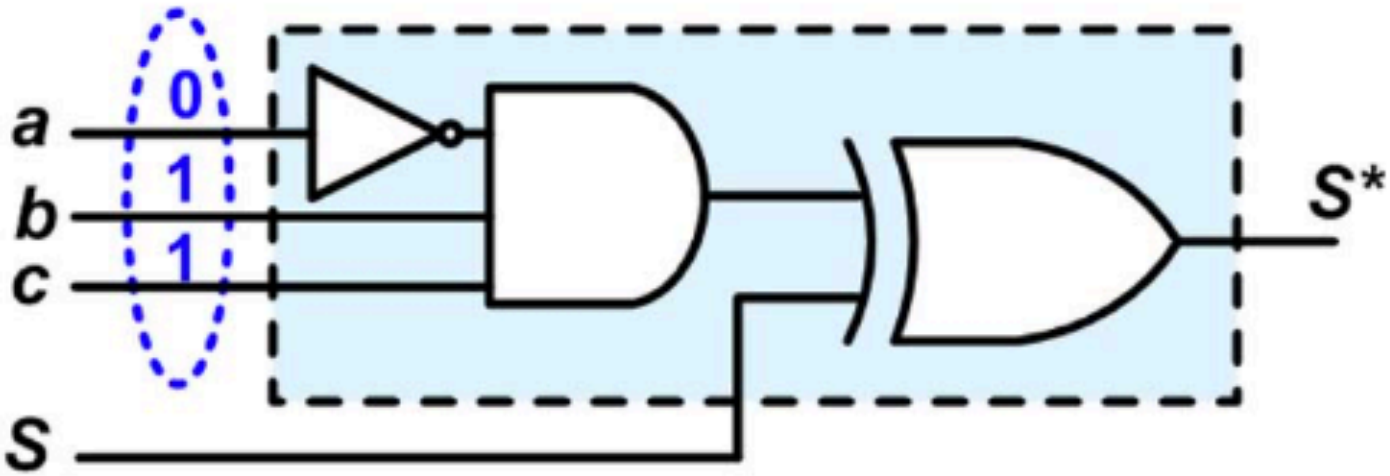


Detection: SCA Analysis [7]

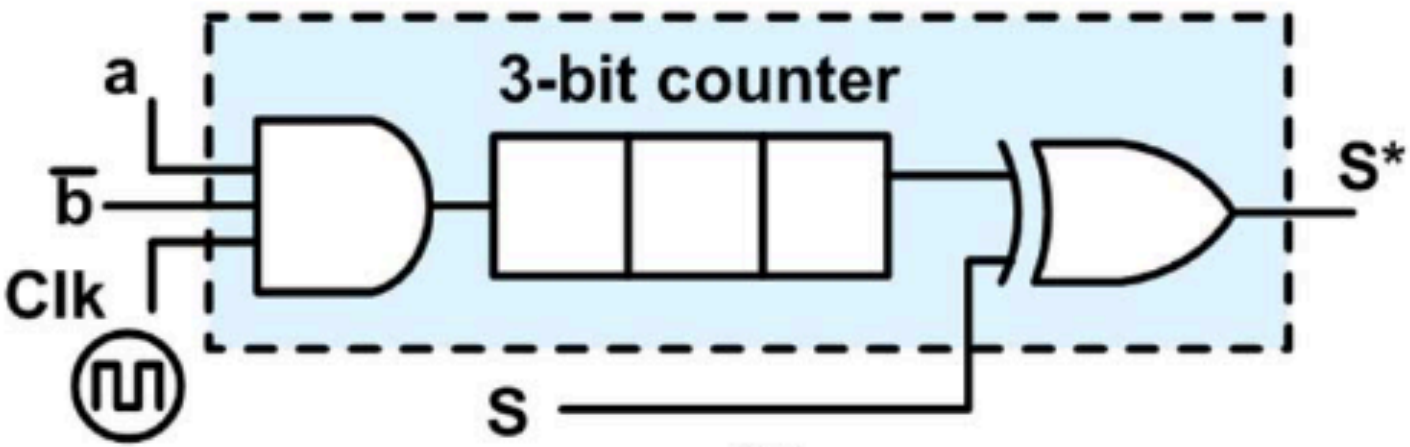
1. HT designed to reduce signature: area, delay, power
2. VLSI increases process variations
3. Use statistical analysis



Invasive HT Detection: Excitation of Rare Event Trigger



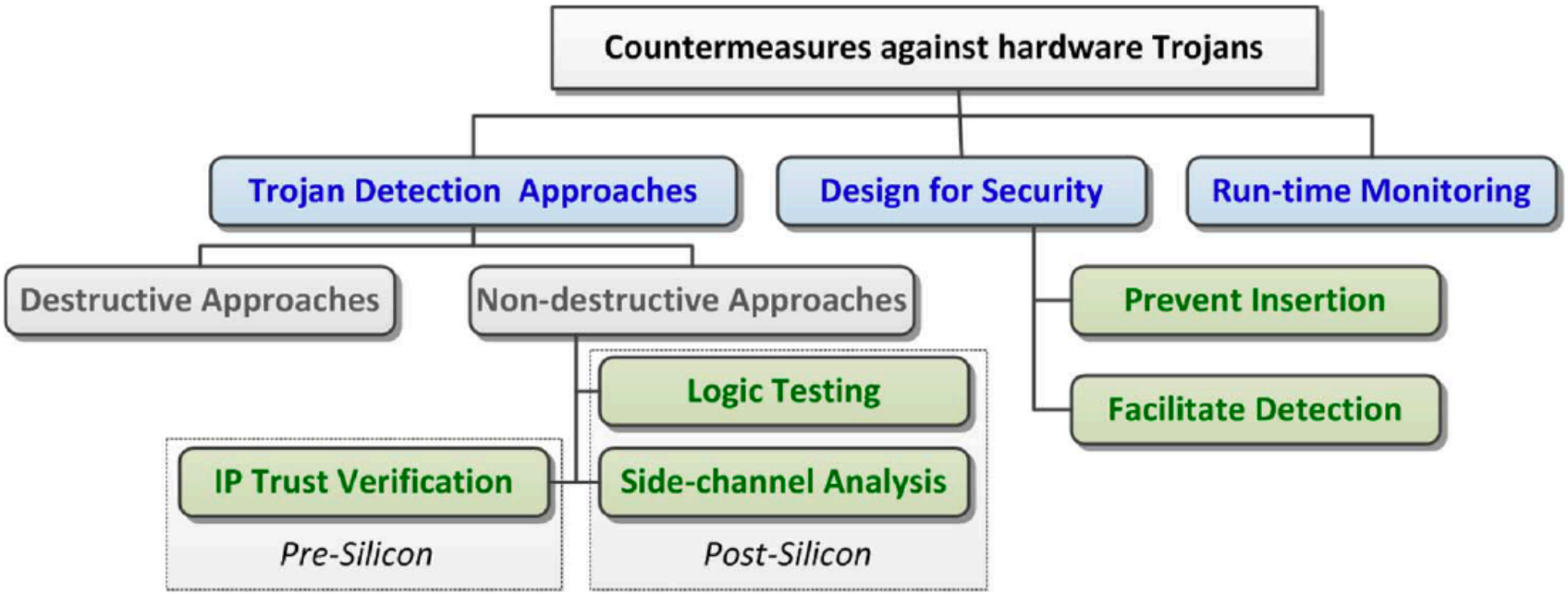
(a)



(b)

HT Countermeasures

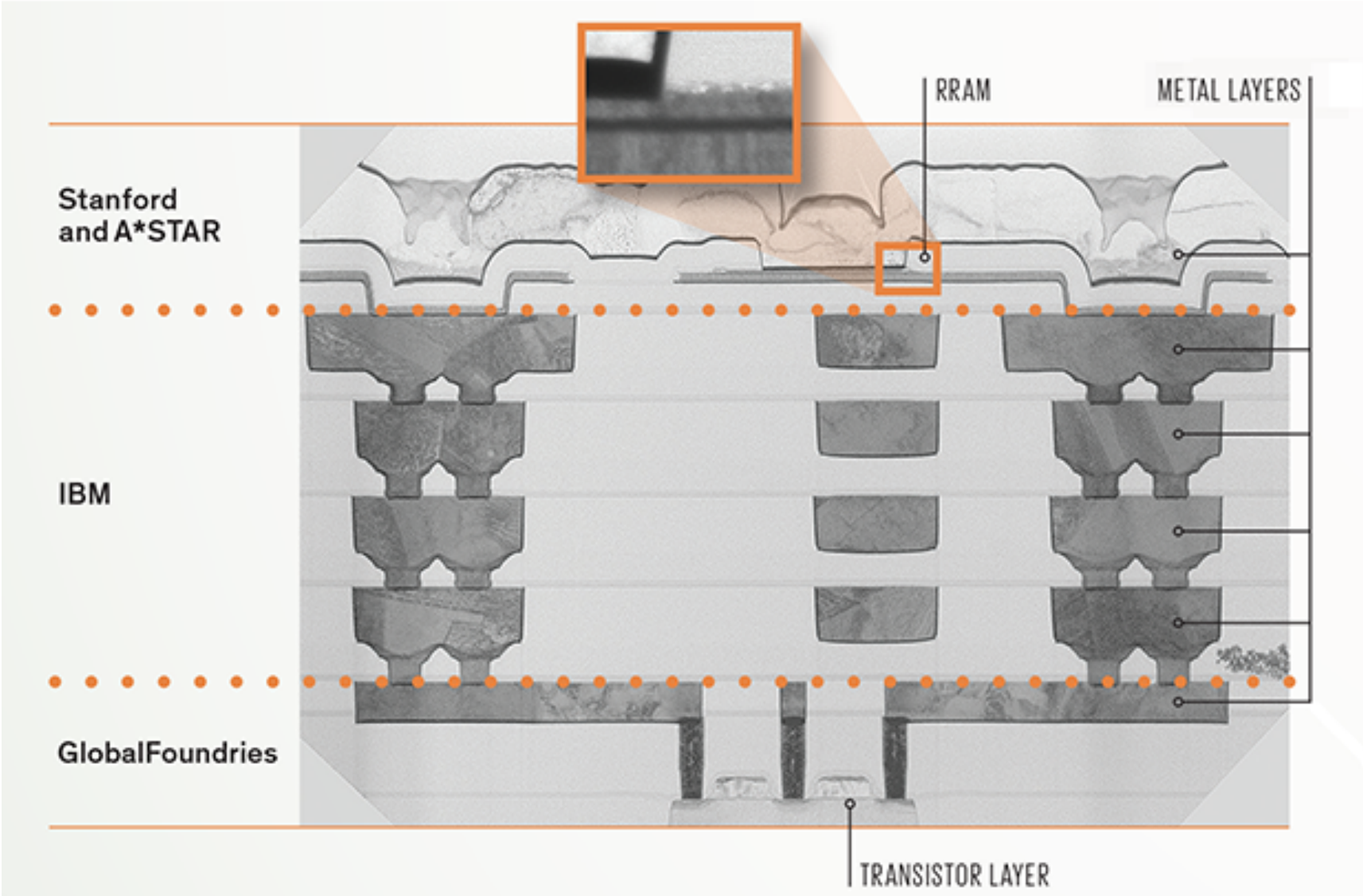
Trojan Countermeasures



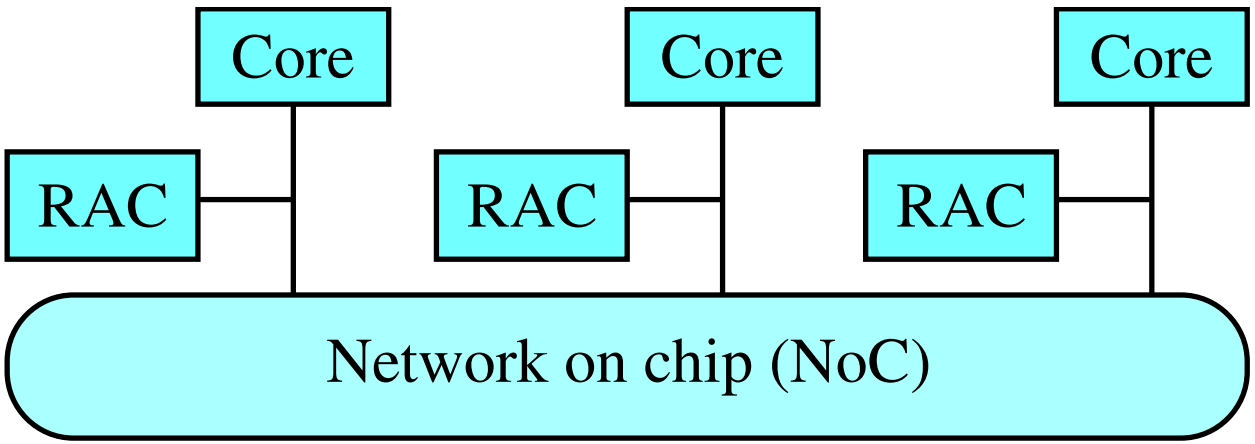
Countermeasures: Split Manufacturing [21]

1. Utilize complexity of the design to defeat Trojan insertion.
2. Delegate "front-end" fabrication to an untrusted state of the art foundry
3. Do the back-end fabrication at a trusted low-tech foundry

Countermeasures: Split Manufacturing



Countermeasures: Using Assertion Checkers [12]



RAC: reconfigurable assertion checker

[1] S. K. Haider, C. Jin, and M. van Dijk, “Advancing the state-of-the-art in hardware trojans design,” *Computing Research Repository (CoRR)*, vol. abs/1605.08413, 2016. [Online]. Available: http://arxiv.org/abs/1605.08413

[2] L. Lin, M. Kasper, T. Guneysu, C. Paar, and W. Burleson, “Trojan side-channels: Lightweight hardware trojans through side-channel engineering,” in *Crypto-graphic Hardware and Embedded Systems - CHES 2009, LNCS*, 2009, pp. 382–395.

[3] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson, “Stealthy dopant-level hardware trojans,” in *Workshop on Cryptographic Hardware and Embedded Systems (CHES 2013)*, Santa Barbara, California, August 20–23 2013.

[4] S. T. King, J. Tucek, A. Zozzie, C. Grier, W. Jiang, and Y. Zhou, “Designing and implementing malicious hardware,” in *First USENIX Workshop on Large-Scale*

Exploits and Emergent Threats (LEET '08) Botnets, Spyware, Worms, and More, 2008.

[5] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware trojans," *IEEE Computer*, pp. 39–46, Oct. 2010.

[6] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, "Hardware trojan attacks: Threat analysis and countermeasures," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1229–1247, 2014.

[7] C. Krieg, A. Dabrowski, H. Hobel, K. Krombholz, and E. Weippl, *Hardware Malware*. Morgan & Claypool, 2013.

[8] W. Hu, B. Mao, J. Oberg, and R. Kastner, "Detecting hardware trojans with gate-level information-flow tracking," *IEEE Computer*, pp. 44–52, Aug. 2016.

[9] S. Chenoweth, S. Indrakanti, and P. Buckland, "On the effects of an emulated memory trojan on the secure

operation of a firewall," https://pdfs.semanticscholar.org/de9d/ec397946d18339e5db4c64d3ac32450d5775.pdf.

[10] D. Agrawal, S. Baktir, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *IEEE Symposium on Security and Privacy*, 2007.

[11] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis: Leaking secrets," in *CRYPTO'99", volume 1666 of LCNS*, 1999, pp. 388–397. Springer.

[12] U. Alsaiari and F. Gebali, "Hardware trojan detection using reconfigurable assertion checkers," *IEEE Explore*, vol. 27, no. 7, pp. 1575–1586, 2019.

[13] R. S. Wahby, M. Howald, S. Garg, A. Shelat, and M. Walfish, "Verifiable ASICs," in *IEEE Symposium on Security and Privacy*, 2016.

[14] T. Wehbe, V. J. Mooney, and D. C. Keezer, "Work-in-progress: A chip-level security framework for

assessing sensor data integrity," in *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2018.

[15] A. Kulkarni, Y. Pino, and T. Mohsenin, "Adaptive real-time trojan detection framework through machine learning," in *IEEE International Symposium on Hardware Oriented Security and Trust*, 2016.

[16] S. Skorobgatove and C. Woods, "Breakthrough silicon scanning discovers backdoor in military chip," in *Cryptographic Hardware and Embedded Systems Workshop*, 2012.

[17] H. Salmani, M. Tehranipoor, and J. Plusquellic, "New design strategy for improving hardware trojan detection and reducing trojan activation time," in *IEEE International Workshop on Hardware-Oriented Security and Trust*, 2009, pp. 66–73.

[18] J. Li and J. Lach, "At-speed delay characterization for IC authentication and trojan horse detection," in *IEEE International Workshop on Hardware-Oriented Security and Trust*, 2008.

[19] G. Bloom, B. Narahari, and R. Simha, "OS support for detecting trojan circuit attacks," in *IEEE International Workshop on Hardware-Oriented Security and Trust*, 2009.

[20] D. R. McIntyre, F. G. Wolff, C. A. Papachristou, and S. Bhunia, "Dynamic evaluation of hardware trust," in *IEEE International Workshop on Hardware-Oriented Security and Trust*, 2009.

[21] S. Mitra, H.-S. Wong, and S. Wong, "The trojan-proof chip," *IEEE Spectrum*, pp. 47–51, feb 2015.

[22] I. D. A. S. Committee, "Property specification language (PSL)," 2010.

[23] M. Oya, Y. Shi, M. Yanagisawa, and N. Togawa, “In-situ trojan authentication for invalidating hardware-trojan functions,” in *17th International Symposium on Quality Electronic Design (ISQED)*, 2016.

[24] F. Wolff, C. Papachristou, S. Bhunia, and R. S. Chakraborty, “Towards trojan-free trusted ICs: Problem analysis and detection scheme,” in *Design, Automation and Test in Europe*, 2008.