A Survey of Hardware Trojan Taxonomy and Detection

Mohammad Tehranipoor
University of Connecticut

Farinaz Koushanfar
Rice University

Editor's note:
Today's integrated circuits are vulnerable to hardware Trojans, which are malicious alterations to the circuit, either during design or fabrication. This article presents a classification of hardware Trojans and a survey of published techniques for Trojan detection.

—Krish Chakrabarty, Editor in Chief

Because of globalization of the semiconductor design and fabrication process, ICs are becoming increasingly vulnerable to malicious activities and alterations. These vulnerabilities have raised serious concerns regarding possible threats to military systems, financial infrastructures, transportation security, and household appliances. An adversary can introduce a Trojan designed to disable or destroy a system at some future time, or the Trojan could leak confidential information and secret keys covertly to the adversary. Trojans can be implemented as hardware modifications to ASICs, commercial-off-the-shelf (COTS) parts, microprocessors, microcontrollers, network processors, or digital-signal processors (DSPs). They can also be implemented as firmware modifications to, for example, FPGA bitstreams. These concerns have been documented in recent reports from the US Defense Science Board task force,[1] the US Senate,[2] *IEEE Spectrum*,[3] and Semiconductor Equipment and Materials International (SEMI).[4]

An IC fabrication process contains three major steps: *design* (which includes IP, models, tools, and designers); *fabrication* (which includes mask generation, lithography, and packaging), and *manufacturing test*. In an ASIC design process, the chip is commonly designed using tools developed by trusted companies—that is, commercial CAD tool developers such as Synopsys, Cadence Design Systems, Mentor Graphics, and Magma Design Automation. However, the IP blocks, models, and standard cells used by the designer during the design process and by the foundry during the postdesign processes are considered untrusted. The fabrication step might also be considered untrusted, because an attacker could substitute Trojan ICs for genuine ones during transit or could subvert the fabrication process itself by implanting a Trojan into the IC mask. Manufacturing test, if done only in the *production test center* of the client (semiconductor company or government agency), would be considered trusted.

There are two main options to ensure that a chip used by the client is *authentic*—meaning it performs only those functions originally intended and nothing more. The first option is to make the entire fabrication process trusted. This option is prohibitively expensive and nearly impossible with the current trends in the global distribution of the IC design and fabrication steps. The second option is to verify the trustworthiness of the manufactured chips upon return to the clients. This requires defining a postmanufacturing step to validate the chip's conformance with the original functional and performance specifications. We call this new step *silicon design authentication*.

Generally, hardware-based security techniques modify hardware to prevent attacks and to protect IP blocks or secret keys. However, the types of attacks we're concerned with in this article are fundamentally different. Here, the attacker is assumed to maliciously alter the design before or during fabrication. Detection of such alterations is extremely difficult, for several reasons. First, given the large number of soft,

0740-7475/10/$26.00 © 2010 IEEE Copublished by the IEEE CS and the IEEE CASS

firm, and hard IP cores used in SoCs, as well as the high complexity of today's IP blocks, detecting a small malicious alteration is extremely difficult. Second, nanometer IC feature sizes make detection by physical inspection and destructive reverse engineering very difficult and costly. Moreover, destructive reverse engineering does not guarantee that the remaining ICs will be Trojan free, especially when Trojans are selectively inserted into a portion of the chip population.

Third, Trojan circuits, by design, are typically activated under very specific conditions (e.g., connected to low-transition-probability nets or sensing a specific design signal such as power or temperature), which makes them unlikely to be activated and detected using random or functional stimuli. Fourth, tests used to detect manufacturing faults such as stuck-at and delay faults cannot guarantee detection of Trojans. Such tests operate on the netlist of a Trojan-free circuit and therefore cannot activate and detect Trojans. Even when 100% fault coverage for all types of manufacturing faults is possible, there are no guarantees as far as Trojans are concerned. Finally, as physical feature sizes decrease because of improvements in lithography, process and environmental variations have an increasingly greater impact on the integrity of the circuit parametrics. Thus, detection of Trojans using simple analysis of these parametric signals would be ineffective.

Hardware Trojans are modifications to original circuitry inserted by adversaries to exploit hardware or to use hardware mechanisms to gain access to data or software running on the chips. Hardware Trojan detection is still a fairly new research area, but it has gained significant attention in the past few years. This survey presents the current state of knowledge on existing detection schemes and design methodologies for improving Trojan detection techniques. We discuss attempts at developing hardware Trojans in IP cores and ICs. We also describe existing Trojan detection methods, analyze their effectiveness in disclosing various types of Trojans, and demonstrate several architecture-level solutions. Finally, we summarize design methods to improve detection techniques' sensitivity to Trojans.

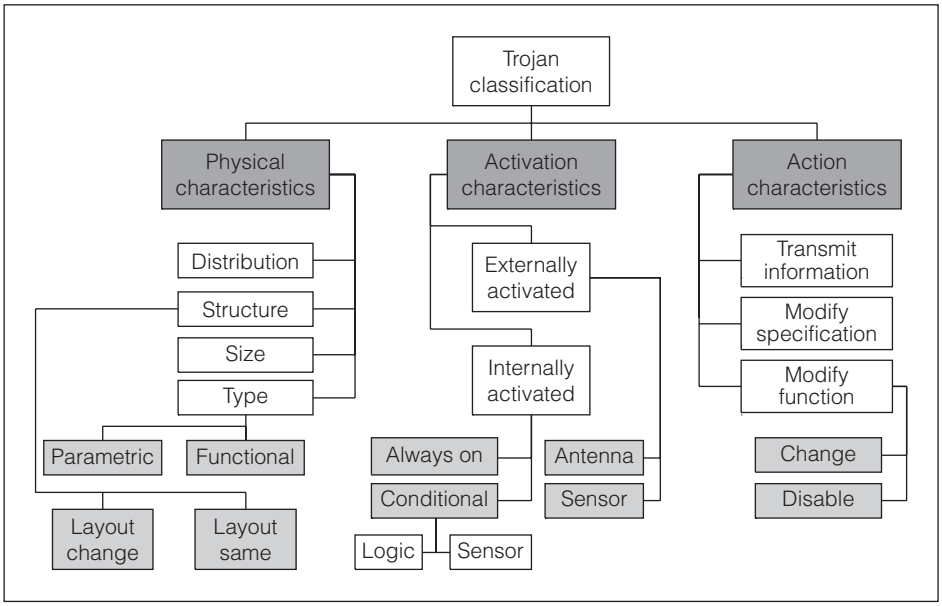

Figure 1. Detailed taxonomy showing physical, activation, and action characteristics of Trojans. (Source: Wang et al.[5])

Trojan design and taxonomy

Wang, Tehranipoor, and Plusquellic developed the first detailed taxonomy for hardware Trojans[5] (a simple taxonomy devised earlier differentiated between payload activation logic and triggering[6]). This comprehensive taxonomy lets researchers examine their methods against different Trojan types.[5] Currently, the industry lacks metrics to evaluate the effectiveness of methods in detecting Trojans. Such metrics could foster a comprehensive taxonomy to help analyze Trojan detection techniques. Because malicious alterations to a chip's structure and function can take many forms, Wang and colleagues decomposed the Trojan taxonomy into three main categories (see Figure 1) according to their physical, activation, and action characteristics. Although Trojans could be hybrids of this classification (for instance, they could have more than one activation characteristic), this taxonomy captures the elemental characteristics of Trojans and is useful for defining and evaluating the capabilities of various detection strategies.

The *physical* characteristics category describes the various hardware manifestations of Trojans. The *type* category partitions Trojans into functional and parametric classes. The *functional* class includes Trojans

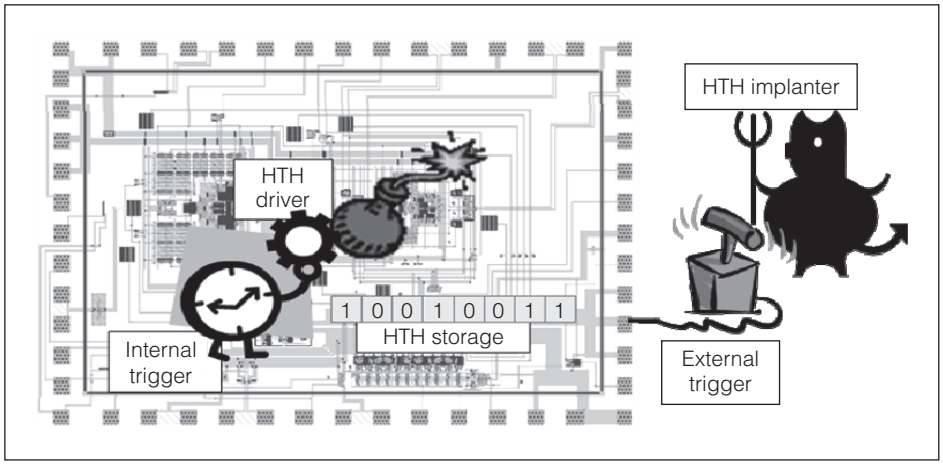

Figure 2. Three components of a hardware Trojan horse (HTH). (Source: Alkabani and Koushanfar[7])

that are physically realized through the addition or deletion of transistors or gates, whereas the *parametric* class refers to Trojans that are realized through modifications of existing wires and logic. The *size* category accounts for the number of components in the chip that have been added, deleted, or compromised. The *distribution* category describes the location of the Trojan in the chip's physical layout. The *structure* category refers to the case when an adversary is forced to regenerate the layout to insert a Trojan, which could then cause the chip's physical form factor to change. Such changes could result in different placement for some or all design components. Any malicious changes in physical layout that could change the chip's delay and power characteristics would facilitate Trojan detection. Wang and colleagues identified current adversaries' capabilities for minimizing the probability of detection.[5]

Activation characteristics refer to the criteria that cause a Trojan to become active and carry out its disruptive function. Trojan activation characteristics fall into two categories: *externally activated* (e.g., by an antenna or a sensor that can interact with the outside world) and *internally activated* (which are further classified as *always on* and *condition based*), as Figure 1 shows. "Always on" means the Trojan is always active and can disrupt the chip's function at any time. This subclass covers Trojans that are implemented by modifying the chip's geometries such that certain nodes or paths have a higher susceptibility to failure. The adversary can insert the Trojans at nodes or paths that are rarely exercised. The condition-based subclass includes Trojans that are inactive until a specific condition is met. The activation condition could be based on the output of a sensor that monitors temperature, voltage, or any type of external environmental condition (such as electromagnetic interference, humidity, altitude, or temperature). Alternatively, this condition could be based on an internal logic state, a particular input pattern, or an internal counter value. The Trojan in these cases is implemented by adding logic gates and/or flip-flops to the chip, and hence is represented as a combinational or sequential circuit.

Action characteristics identify the types of disruptive behavior introduced by the Trojan. The classification scheme shown in Figure 1 partitions Trojan actions into three categories: modify function, modify specification, and transmit information. The modify-function class refers to Trojans that change the chip's function by adding logic or by removing or bypassing existing logic. The modify-specification class refers to Trojans that focus their attack on changing the chip's parametric properties, such as delay when an adversary modifies existing wire and transistor geometries. Finally, the transmit-information class includes Trojans that transmit key information to an adversary. (Additional details on Trojan classification and examples are available elsewhere.[5])

Researchers have designed many types of Trojans to evaluate their detection techniques by targeting them in an IC.[6-10] To imitate adversaries' Trojan insertions, Alkabani and Koushanfar classified the components needed for a hardware Trojan horse (HTH) into three categories: trigger, storage, and driver (see Figure 2).[7] A trigger incites the planned HTH. After a trigger occurs, the action to be taken can be stored in memory or a sequential circuit. A driver implements the action prompted by the trigger. On the basis of the classification just described, Alkabani and Koushanfar present a systematic approach to insert hardware Trojans into the IC using presynthesis manipulation of the circuit's structure.[7] Such a model addresses the issue of trust in IP cores designed by either a third-party vendor or a system integrator when several IP cores developed by many vendors are used.

Figure 3 shows an abstracted view of the design process. The Trojan designer composes the high-level design description to find the computation model of the circuit that a finite-state machine (FSM) can represent. An HTH can be inserted into the circuit by altering the FSM and embedding states into it. The modified FSM should have a trigger as an input and a driver hidden in the structure of the FSM. This FSM can be systematically hidden in the design by merging its states within the states of the original design's FSM. Thus, the HTH would be inseparable (unremovable) from the original design's functionality. A stealth communication, which uses the medium for legitimate communications, can serve as a covert channel to transfer confidential data from the working chips to the adversary. This Trojan-embedding approach provides a low-level mechanism for bypassing higher-level authentication techniques.

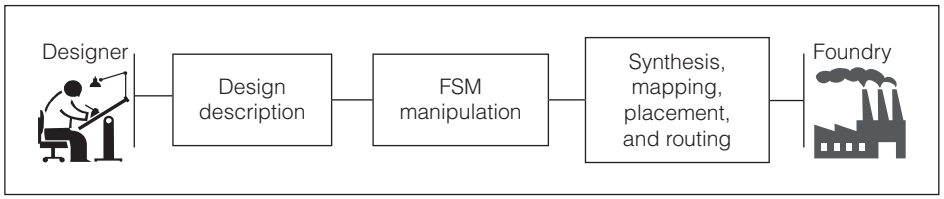

Figure 3. Insertion of an HTH during the design process of an IP core. (Source: Alkabani and Koushanfar[7])

Jin, Kupp, and Makris investigated different types of attacks on a design at the RTL.[8] Specifically, they examined the possibility of designing hardware Trojans that can evade state-of-the-art detection methodologies and can pass functional test.

King et al. considered the malicious circuit design space and introduced hardware components that can enable several attacks.[11] In particular, they designed and implemented the Illinois Malicious Processor with a modified CPU. The malicious modifications allow memory access and shadow-mode mechanisms. The former lets an attacker violate operating-system isolation expectations, whereas the latter admits stealthy execution of malevolent firmware. The attacks were evaluated on an FPGA development board by modifying the VHDL code of the Leon processor, an open-source Sparc v8 processor that includes a memory management unit. The overhead in logic is less than 1% for both modifications, but the timing overhead is about 12%. The authors further designed and implemented three potential attacks: a privilege escalation attack, which gives an intruder access to the root without checking credentials or generating log entries; a log-in backdoor in shadow mode, which lets an intruder log in as a root without using a password; and a service for stealing passwords and sending them to the attacker. They concluded that hardware tampering is practical and could support various attacks, while also being difficult to detect.

Mechanisms for actively controlling an IC can also be used insert a malicious circuit in a design (i.e., IP core). For example, manipulation of the states in an FSM that cannot be reverse-engineered could be used to embed Trojan circuitry by providing mechanisms for remotely activating, controlling, and disabling the Trojan.[12]

Trojan detection methodologies

Several Trojan detection methodologies have been developed over the past few years. Without loss of generality, the methods are categorized as either *side-channel analysis* or *Trojan activation*, which are mainly chip-level solutions and architectural-level Trojan detection solutions.

Trojan detection using side-channel signal analysis

Side-channel signals, including timing and power, can be used for Trojan detection. Trojans typically change a design's parametric characteristics—for example, by degrading performance, changing power characteristics, or introducing reliability problems in the chip. This influences power and/or delay characteristics of wires and gates in the affected circuit. Power-based side-channel signals provide visibility of the internal structure and activities within the IC, enabling detection of Trojans without fully activating them. Timing-based side channels can detect a Trojan's presence if the chip is tested using efficient delay tests that are sensitive to small changes in the circuit delay along the affected paths and that can effectively differentiate Trojans from process variations.

Power-based analysis. Agrawal et al. were the first to use side-channel information to detect Trojan contributions to circuit power consumption.[13] To obtain the power signature of Trojan-free (i.e., genuine)

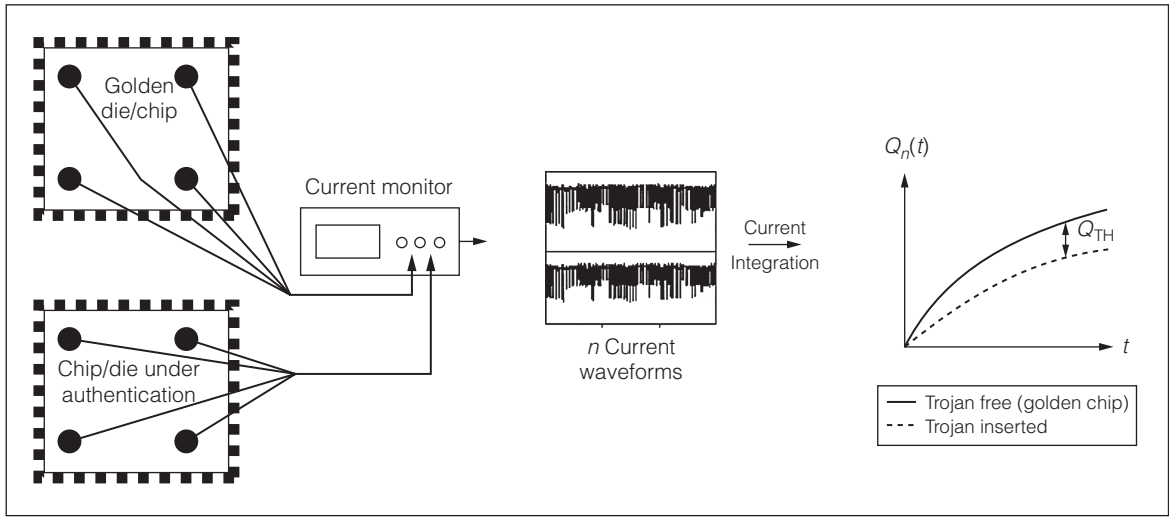

Figure 4. Current (charge) integration method. (Source: Wang et al.[9])

ICs, random patterns are applied and power measurement is performed. The data belonging to each power measurement consists of several elements, including power consumption of the circuit after applying inputs that are the same in all Trojan-free ICs; measurement noise, which can be removed by several measurements; process variations, which are random and cannot be removed; and Trojan contributions to the measured power consumption. After patterns are applied, a limited number of ICs are reverse-engineered to ensure they are Trojan free. Once the reference signature is obtained, the same random patterns are applied to the IC under authentication (IUA). If the IUA's power signature differs from the reference signature, the IUA is considered suspicious and that it might contain a Trojan. Trojans of different sizes under different process variations are detected by applying random patterns and observing the signatures. If the Trojan is comparable in size with the circuit, its impact on the circuit-transient current will be significant and could be measured easily. However, process variations will mask the impact of very small Trojans on circuit power consumption.

Wang and colleagues argued that most Trojans inserted into a chip require power supply and ground to operate.[9] The Trojans can be of different types and sizes, and their impact on circuit power characteristics could be very large or very small. The authors developed a multisupply transient-current integration methodology to detect a hardware Trojan. Then, they introduced a Trojan isolation method based on localized-current analysis. They assumed the current is measured from various power ports or controlled collapse chip connections (C4s) on the die, and they applied random patterns to increase the switching in the circuit in a test-per-clock fashion.

The amount of current that a Trojan can draw might be so small that it could be submerged into an envelope of noise and process variation effects, and thus be undetectable by conventional measurement equipment. However, Trojan detection capability can be greatly enhanced by measuring currents locally and from multiple power ports or pads. Figure 4 shows the current (charge) integration methodology presented by Wang et al. for detecting hardware Trojans.[9] The die includes four power ports. The golden die can be identified using an exhaustive test for several randomly selected dies. It can also be identified via the pattern set used in the current integration method by comparing the results of all patterns in an exhaustive fashion. If the same results (within the range of variations) are obtained for all selected dies, those dies can be identified as Trojan free.

The authors assumed the adversary will insert the Trojans randomly in a selected number of chips.[9] After the golden dies are identified, the worst-case charge is obtained (dashed line in Figure 4 in response to the pattern set. The worst-case charge is based on the worst-case process variations in one of the genuine ICs. Next, the pattern set is applied to each chip, and the current is measured for each pattern locally via the power ports or C4 bumps.

Figure 4 shows the current waveform of n number of patterns applied to the chips. The figure also illustrates the charge variations with time for all the current waveforms obtained after applying the patterns.

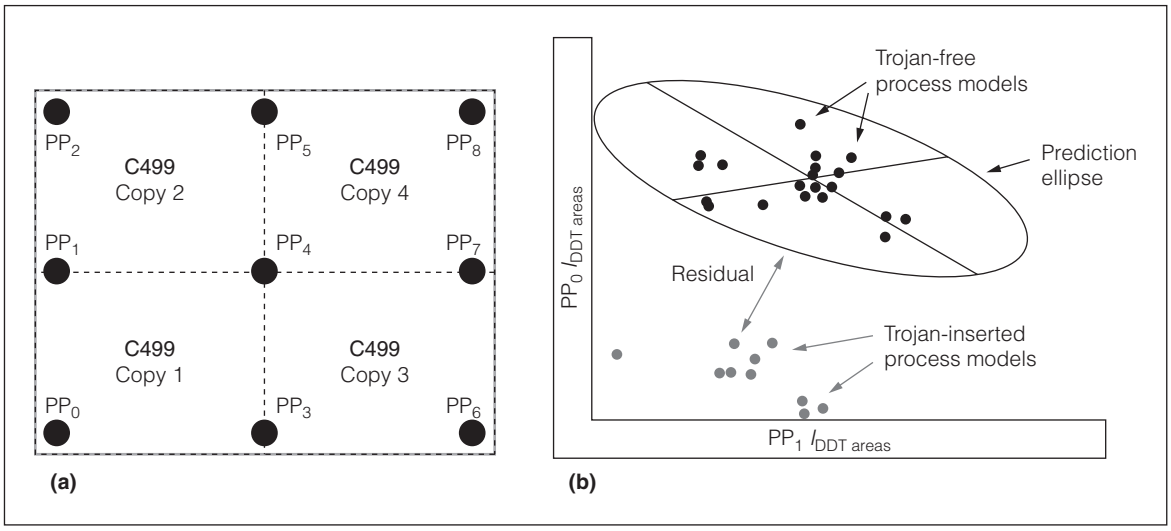

Figure 5. Architecture of the simulation model used by Rad et al. (a), and an example scatter plot for PP01 (power ports 0 and 1) from one of the experiments (b). (Source: Rad et al.[10])

The charge corresponds to the area produced by each current waveform. $Q_n(t)$ denotes the accumulative charge after applying n patterns. Q_{TH} is the charge threshold to detect a Trojan; this threshold is the resolution measurement defined by the instrumentation. When the patterns are applied, the charge increases and is compared continuously against the worst-case charge calculated for golden dies. Once the difference between the two curves ΔQ is greater than Q_{TH}, a Trojan is detected. The number of patterns n is expected to be very small for large Trojans, and large for very small Trojans. The application time is expected to be low, because the patterns are applied in a test-per-clock fashion.

Rad et al. proposed a region-based transient-power signal analysis method to reduce the impact of increasing process variation levels and leakage currents.[10] A *region* is a portion of the layout that receives the majority of its power from surrounding power ports or C4 bumps. Figure 5a shows a six-metal-layer power grid with nine power ports. Measurements are made through each power port individually by applying patterns. The transient-current detection algorithm is based on a statistical analysis of the I_{DDT} waveform areas generated at the nine power ports as a test sequence is simulated on the design. For each orthogonal pairing of power ports, a scatter plot is constructed. The authors used several different process models for Trojan-free and Trojan-inserted designs. Figure 5b shows that using a prediction ellipse derived from a Trojan-free design with different process models can help distinguish between Trojan-inserted and Trojan-free designs. The dispersion in the Trojan-free data points is a result of uncalibrated process and test environment (PE) variations.

However, regional analysis alone is not sufficient for dealing with the adverse effects of PE variations on detection resolution. Signal calibration techniques are necessary to attenuate and remove PE signal variation effects, to fully leverage the resolution enhancements available in a region-based approach. Calibration is performed on each power port and for each chip separately, and it measures the response of each power port to an impulse. The response of each power port X (PP_X) is normalized by the sum of current drawn from power ports in the same row as PP_X. The calibration matrix comprises the normalized values of power ports. After each test pattern is applied, the response is calibrated using the calibration matrix. The results presented by Rad et al. show that calibration can increase the distance between Trojan-free and Trojan-inserted designs (see residual in Figure 5b) under different process parameters.

Recently, Alkabani and Koushanfar proposed several approaches for gate-level timing and power characterization via nondestructive measurements.[14] Each measurement forms one equation. After a linear number of measurements are taken, a system of equations for mapping the measured characteristics to the gate level is formed.

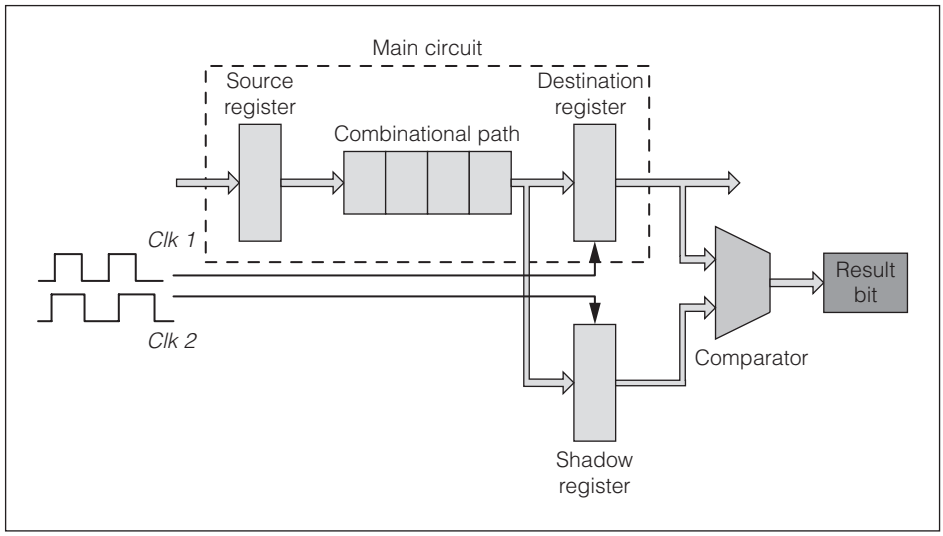

Figure 6. Path delay measurement architecture using a shadow register. Such an architecture can be used for IC authentication and Trojan detection. (Source: Li and Lach[16])

Potkonjak et al. exploited the formulation of gate-level characterization using linear programming and singular-value decomposition to detect Trojans.[15] They used both timing and static-power measurements. Trojan detection is performed via constraint (equation) manipulation. This method attempts to find the measurement matrix with the highest rank, and derives several heuristics for detecting gates that have inconsistent characteristics compared to their original specified characteristics. Learn, test, and resubstitution statistical validation techniques are used to estimate the bounds for normal (nonmalicious) characteristics. The experiments considered errors in noninvasive measurements, but not process variations. The evaluation results are promising because gate-level characterization with high accuracy is possible. The gate-level characterization methods can find the characteristics of controllable gates. This controllability is known to be high for static-power measurements and I_{DDQ} testing.

Alkabani and Koushanfar used statistical convergence of gate-level estimation and signal integrity for Trojan detection.[14] They found efficient robust approximations for gate power consumptions and identified malicious insertions using multiple consistency checking.

Timing-based analysis. Li and Lach proposed a delay-based physical unclonable function (PUF) for hardware Trojan detection.[16] This method uses a sweeping-clock-delay measurement technique to measure selected register-to-register path delays. Trojans can be detected when one or a group of path delays are extended beyond the threshold determined by the process variations level. Figure 6 shows the path delay measurement architecture. The main circuit is the register-to-register combinational path that is to be characterized, and the registers on this path are triggered by the main system clock (*CLK1*). The components outside the box are part of the testing circuitry. The *shadow register* takes the same input as the destination register in the main circuit but is triggered by the shadow clock (*CLK2*), which runs at the same frequency as *CLK1* but at a controlled phase offset. The results latched by the destination register and the shadow register are compared during every clock period. If the comparison result is unequal, the path delay is characterized with a precision of the skew step size.

This method employs an on-die temperature monitor to overcome the problem of temperature affecting path delay. This monitor uses a ring oscillator as the clock input of a counter to measure operating temperature. Because the oscillator is embedded within the main circuitry and its switching frequency is temperature dependent, the authenticator can calculate the effective response from the reported temperature and delay signature. Although effective, this technique suffers from considerable area overhead when targeting today's large designs with millions of paths.

Jin and Makris proposed a new fingerprint-generating method using path delay information of the entire chip.[17] A chip has many delay paths, each representing one part of the characteristic of the entire chip. The timing features can generate a series of path delay fingerprints. Regardless of how small the Trojan is compared to the entire chip size, it can be significant in the path view and could be detected. The entire testing procedure includes three steps:

1. *Path delay gathering of nominal chips.* Many chips are selected from a fabricated design.

High-coverage input patterns are run on the sample chips, and high-dimension path delay information is collected. Then, the sample chips are checked via reverse engineering to ensure they are genuine circuits.

2. *Fingerprint generation*. According to the path delays, a series of delay fingerprints are generated and mapped to a lower-dimension space.
3. *Trojan detection*. All other chips are checked under the same test patterns. Their delay information is reduced to a low dimension and compared to the delay fingerprints.

This method uses statistical analysis to deal with process variations. Because today's circuits can include millions of paths, measuring all paths—especially the short ones—is not practical.

Trojan activation methods

Trojan activation strategies can accelerate the Trojan detection process, and in some cases have been combined with power analysis during implementation. If a portion of the Trojan circuitry is activated, the Trojan circuit will consume more dynamic power, which will further help differentiate the power traces of Trojan-inserted and Trojan-free circuits. The existing Trojan activation schemes can be categorized as follows.

Region-free Trojan activation. These methods do not rely on the region but depend on accidental or systematic activation of Trojans. For example, Jha and Jha presented a randomization-based probabilistic approach to detect Trojans.[18] They showed that it's possible to construct a unique probabilistic signature of a circuit on the basis of a specific probability for patterns applied to its inputs. They apply input patterns based on the specific probability to IUA and compare its outputs with the original circuit. If there are differences in the outputs, a Trojan is present. For Trojan detection in a manufactured IC, patterns can be applied only on the basis of such probability to obtain a confidence level regarding whether the original design and the fabricated chip are the same.

Wolff et al. analyzed rare-net combinations in designs.[6] These rarely activated nets are used as Trojan triggers. At the same time, nets with low observability are used as payloads (see Figure 7). Wolff et al. generated a set of vectors to activate such nets and suggested combining them with traditional ATPG test vectors to activate a Trojan and to propagate its impact if the Trojan was connected to these nets.

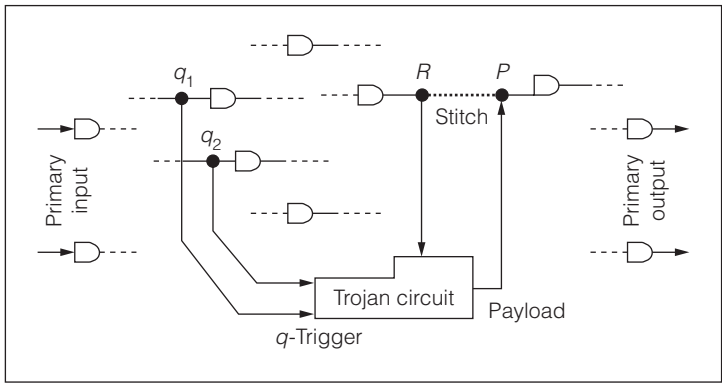

Figure 7. Trojan circuit model with a rare triggering condition. (Source: Wolff et al.[6])

Region-aware Trojan activation. Banga and Hsiao developed a two-stage test generation technique that targets magnifying the difference between the IUA and the genuine design power waveforms.[19] In the first stage (*circuit partitioning*), a region-aware pattern helps identify the potential Trojan insertion regions. To detect a Trojan circuit, the activity within a portion of the circuit is increased while the activity for the rest of the circuit is simultaneously minimized. The flip-flops in a circuit are classified into different groups, depending on structural connectivity. In the next stage (*activity magnification*), new test patterns concentrating on the identified regions are applied to magnify the disparity between the original and Trojan-inserted circuits. Regions (a set of flip-flops) exhibiting increased relative activity are identified by using the vector sequence generated in the first stage to compare the relative differences between the power profiles of the genuine and Trojan circuits. In this stage, more vectors for the specific regions, marked as possible Trojan regions, are generated using the same test generation approach as in the circuit-partitioning stage.

Banga and Hsiao discussed magnifying Trojan contributions by minimizing circuit activity.[20] This involves keeping input pins unchanged for several clock cycles. Thus, circuit activity comes from the state elements of the design. Overall switching activity is therefore reduced, and can be limited to those specific portions of the design that help Trojan localization. Different portions of the design can be explored by changing input vectors to localize a Trojan. At the same time, each gate is equipped

with two counters: *TrojanCount* and *NonTrojanCount*. With each vector, if the number of transitions at a gate's output exceeds a specific threshold, its TrojanCount would increase, and vice versa. The TrojanCount/NonTrojanCount ratio, called the *gate weight*, indicates a gate's activity. A high gate-weight ratio means the gate is considerably impacted by a Trojan, because there is a relatively high power difference corresponding to that gate's activation.

Because the test engineer does not know the Trojan type or size, both region-free and region-aware methods are necessary. If a Trojan circuit's inputs come from the part of the circuit where they are functionally dependent (i.e., part of the same logic cone), the region-aware method can be effective. However, if the Trojan inputs are randomly selected from various parts of the circuit, region-free methods could increase the probability of detection.

Architecture-level Trojan detection

Verbauwhede and Schaumont explored trust issues at different levels of design abstraction (protocols, software, microarchitecture, and circuits).[21] At the most abstract level, the adversary can access the interpreter and perform software tempering, scan-chain readout, or a fault attack. Side-channel information can be used at the software-architecture level. At the hardware microarchitecture and circuit levels, the attacker takes into account power energy consumption or electromagnetic energy. Hence, the authors proposed a systematic countermeasure to protect the root of trust at different design abstractions.

Tamper-proof techniques such as placing security parts into special casing with light, temperature, tampering, or motion sensors can provide protection at the physical level. Side-channel information such as power consumption should be separated from processing data or execution time to provide circuit-level protection. To deal with power fluctuation, different technologies such as full-custom dynamic and differential logic styles should be used. In experiments conducted by the authors, advanced encryption standards employing wave dynamic and differential logic remained safely after 1.5 million power-differential attack measurements, whereas standard CMOS technology disclosed the key only after 2,000 attack measurements.

To deal with side-channel attacks at the microarchitecture level, Verbauwhede and Schaumont suggested balancing if-and-else instructions to use the same amount of time and power during execution. The structure of microprocessors providing potential sources of side-channel information should be considered seriously. The authors also suggested using secure algorithm techniques, such as key and exponent blinding, to disable side-channel attacks at lower levels.

Suh, Deng, and Chan proposed authenticating the hardware by directly checking its implementation details at a low level.[22] The microarchitecture features of a high-end secure microprocessor are complex and unique for each model. A secure processor is authenticated by a checksum response to a challenge within a time limit. The unique checksum is based on the cycle-to-cycle activities of the processor's specific internal microarchitectural mechanism. Privacy is not breached, because the checksum depends on the processor-manufactured model and not the specific processor. The authors showed that small differences in the crypto-architecture result in significant deviations in the checksum. Their work relied on the speed advantages of the actual processor rather than simulations that attempt to impersonate the processor. The time limit on the authentication ensures resiliency against simulation models attempting to compute the checksum.

Bloom, Narahari, and Simha introduced a runtime Trojan activity detection mechanism using a hardware guard circuit and operating-system support.[23] Trojan attacks can either be internally or externally activated, and they can cause denial of service, privilege escalation, or leakage of sensitive information. Trojans can be detected by failure analysis and hardware verification, ATPG, or side-channel analysis. Bloom, Narahari, and Simha's work concentrated on denial-of-service (DoS) and privilege escalation attacks.[23] They used a hardware guard circuit to efficiently perform the testing, while the operating system generated the checks. Their hardware circuit included a timer, a scratch RAM, a simple processor, and an optional content-addressable memory (CAM).

Two tests were proposed: liveness checks and memory protection checks. Liveness checks are pseudorandom noncached-memory accesses that prevent simple prediction, delay, and replay attacks. Two solutions were provided for memory protection: a naive solution and a solution using a real-time operating system (RTOS). The naive solution periodically schedules a process that continuously tries to read the kernel

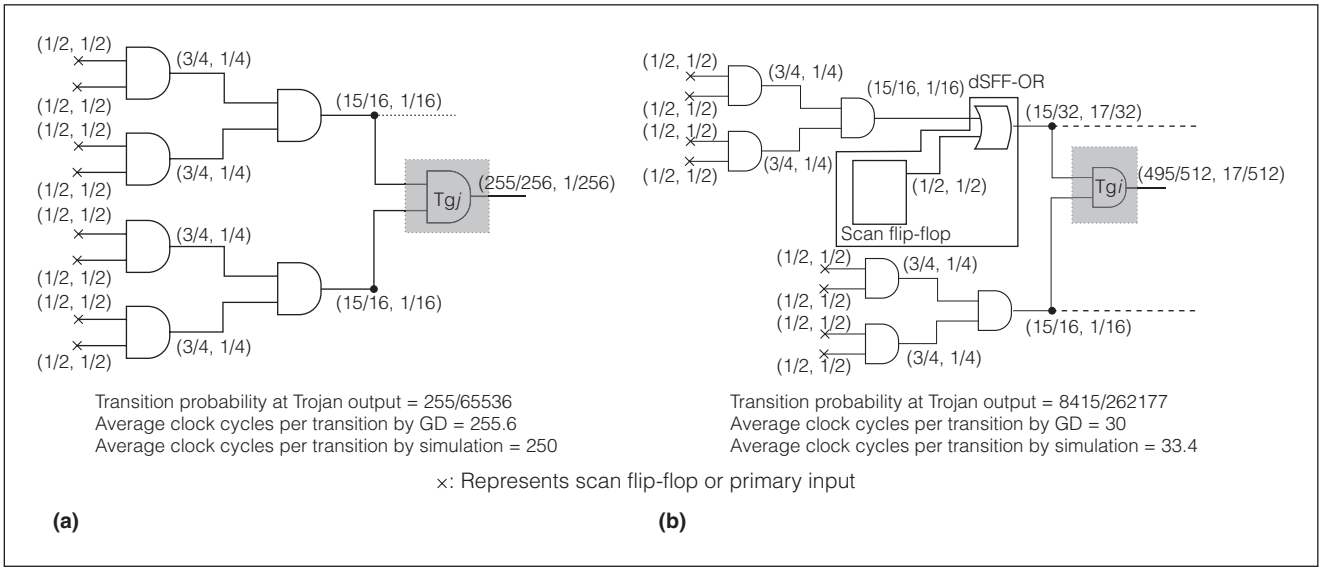

Figure 8. Analyzing transition probability in the original circuit (a) and after dummy scan flip-flop insertion (b). (Source: Salmani et al.[25])

memory. However, the process is time-consuming. RTOS support is needed to control the time of the checking process, which is created as a real-time task that is frequently required and consumes less time. The proposed solutions are evaluated on SPECit 2006 benchmarks. The overhead for using RTOS support is approximately 2.2%.

McIntyre et al. used hardware multicore systems, which permit simultaneous execution of the same functionality combined with verification.[24] Multicore systems are inherently redundant. Thus, as trust detection among the multiple cores is discovered, distributed software scheduling could be exploited to avoid low-trust cores. The distributed multicore task scheduler determines, over time and in the field, each core's hardware trust level.

Design for hardware trust

Current design practices do not support effective side-channel signal analysis or pattern generation for Trojan detection. The CAD and test community has long benefited from DFT and DFM (design for manufacturability). Here, we look closely at some of the methods proposed by the hardware security and trust community to improve Trojan detection and isolation by changing or modifying the design flow. We call these methods *design for hardware trust*. These methods help prevent the insertion of Trojans, facilitate easier detection of Trojans, and provide effective IC authentication.

Salmani, Tehranipoor, and Plusquellic developed a methodology to increase the probability of generating a transition in functional Trojan circuits and to analyze transition generation time.[25] They modeled transition probability using geometric distribution and estimated it on the basis of the number of clock cycles needed to generate a transition on a net. They presented an efficient dummy flip-flop insertion procedure to increase the transition probability of nets when it is lower than a specific probability threshold. Figure 8a shows a circuit with Tg_j as a Trojan gate. The transition probability at the gate output is extremely low. However, after adding dummy scan flip-flops (see Figure 8b) to a net with a low transition probability, the transition probability at the Trojan output increased considerably; similarly, the average number of clock cycles per transition decreased.

Dummy flip-flops are inserted such that they do not alter the design's functionality or performance. To examine the effectiveness of dummy flip-flop insertion, the authors evaluated different transition probability thresholds for various Trojan circuits. They studied in detail the relationships among authentication time, the number of required transitions in the Trojan circuit, and the tester clock. These parameters can help determine a design's transition probability threshold. The transition probability threshold, in turn, provides an estimation of the area overhead induced by the insertion of dummy

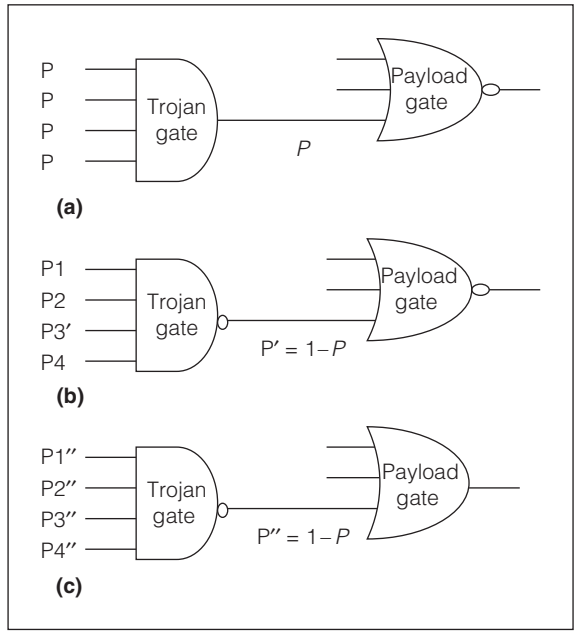

Figure 9. Trojan logic under normal voltage supply (a), where only the Trojan gate is affected by an inverted voltage supply (b), and where both the Trojan and payload gate are affected by an inverted voltage supply (c). (Source: Banga and Hsiao[26])

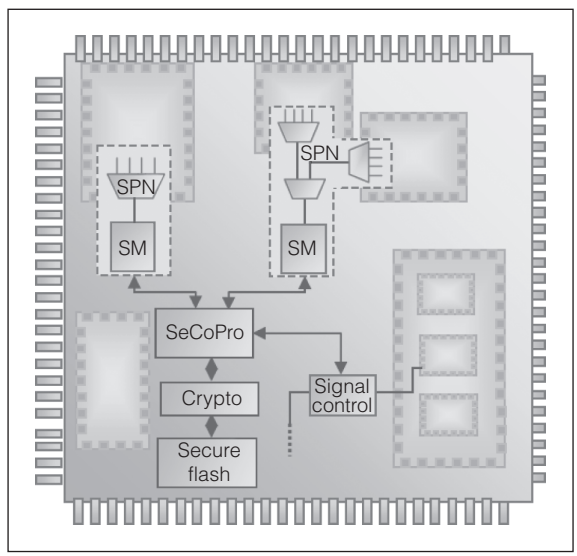

Figure 10. SoC design with design-for-enabling-security (Defense) logic. (Source: Abramovici and Bradley[27])

flip-flops. This method can help detect Trojans in two ways:

- It can improve power-based side-channel signal analysis methods by increasing the number of transitions in Trojans.
- It provides an opportunity for fully activating a Trojan circuit and observing the erroneous response at that circuit's output.

Banga and Hsiao proposed an inverted voltage scheme to magnify Trojan activity.[26] Because the Trojan is assumed to be activated only under rare conditions, IC inputs could be changed so that rare combinations are created to activate the Trojan. For example, for an AND gate with four inputs, a rare condition would be when all its inputs are 1 (a probability of 1/16). The goal is to change the Trojan's functionality to remove the rare condition. Reversing a gate's power supply voltage (V_{DD}) and ground (*GND*) changes its function and reduces the noise margin as the output swings between $V_{DD} - V_{TH}$ and V_{TH} (where V_{TH} is the transistor voltage threshold). Thus, AND changes to NAND, and 1 at the output of a NAND Trojan is no longer a rare value (its probability becomes 15/16, as Figure 9 shows). However, this method must face the difficulty of switching between power supply voltage and ground for each gate on the circuit, because current power distribution networks are not designed to support an inverted voltage scheme.

To monitor an IC's system operation and detect unexpected or illegal behavior, Abramovici and Bradley suggested employing reconfigurable design-for-enabling-security (Defense) logic to the functional design.[27] When an attack is detected, the first step is to deploy countermeasures such as disabling a suspect block or forcing a safe operational mode. Figure 10 shows the architecture of a SoC with Defense inserted. Signal probe networks (SPNs) are configured to select a subset of the monitored signals and transport them to security monitors. A *security monitor* is a programmable transaction engine configured to implement an FSM, to check the current signals' user-specified behavior properties.

The security and control processor (SeCoPro) reconfigures SPNs to select the groups of signals that security monitors must check and reconfigures these security monitors to perform the required checks. All configurations are encrypted and stored in the secure flash memory. The security checks are application dependent and circuit dependent. This approach can detect attacks and prevent the damage they cause, by inserting these security checks into every phase of the design flow. These checks are concealed from attackers trying to reverse-engineer the device,

because the reconfigurable logic is blank (unprogrammed) in a powered-off chip. On the other hand, they are not accessible from either the functional logic or the embedded software. Similarly, SeCoPro is invisible for the other on-chip application processors. However, this approach cannot detect a Trojan unless the chip is fabricated and the method is also unable to locate the Trojan. To increase the efficiency of the method, a large number of nets in the circuit must be selected for observation, and this could increase area overhead.

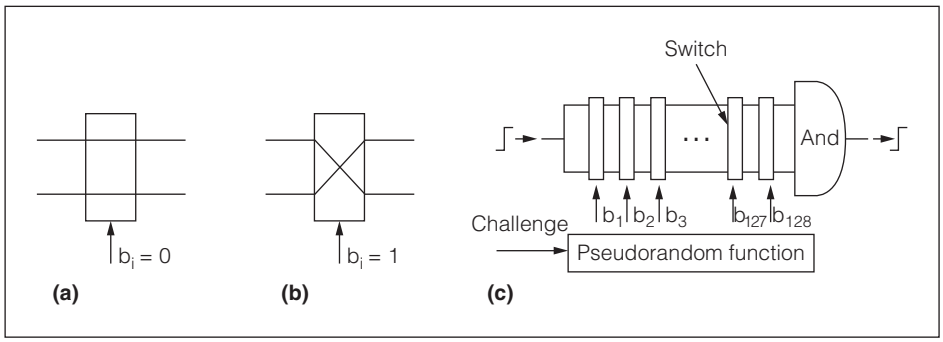

Figure 11. Physical unclonable function (PUF) design 1. Arbiter switch when $b_i = 0$ (a), arbiter switch when $b_i = 1$ (b), and architecture of the first PUF (c). The signature of the PUF is the delay measured by the race between two paths to the AND gate or arbiter. A 128-bit challenge is transformed by a pseudorandom function into a bit pattern /b/ = (/b/_1, ..., /b/_128). (Source: Gassend et al.[30])

Chakraborty, Paul, and Bhunia introduced a design methodology called *on-demand transparency*,[28] which facilitates Trojan detection via logic testing. They defined a special mode of operation, called *transparent mode*, for a multimodule system. In this mode, a signature is generated upon application of a specific input. The hope is that Trojan tampering will affect the signature and reveal the Trojan. The selected nodes are those that are assumed to be most susceptible to Trojan attacks, guided by the controllability and observability values. (This type of signature generation for watermarking circuits and finding watermark tampering had been proposed and used earlier.[29]) The drawback of the authors' proposed method is that an FSM's states contain many don't-care values and an enormous number of states. Thus, generating a signature that each kind of tampering will affect is an extremely challenging problem for larger circuits.

Researchers have attempted to develop on-chip structures to facilitate effective IC identification and authentication, which can help with Trojan detection as well. For example, any change in the circuit physical layout or from moving components in the circuit would potentially change the circuit parasitics parameters and could be detected by on-chip structures.[10,16]

PUFs represent one such structure. Built on many random uncontrollable components originated from process variations, PUFs can reliably and securely identify individual ICs. Gassend et al. proposed two candidate PUF circuits based on delay measurement.[30] Figure 11 shows the architecture of the first PUF. The signature of the PUF is the delay measured by the race between two paths to the AND gate or arbiter. A 128-bit challenge is transformed by a pseudorandom function into a bit pattern $b = (b_1, \ldots, b_{128})$. The bits b_i control switches, as Figures 11a and 11b show. To obtain a response from this circuit, a pulse is applied to its input. The pulse is split into two competing pulses, which independently propagate through the switches until they reach the AND gate. The circuit response is the time it takes for the pulse on the input to produce a pulse on the output. Although measuring delays using self-oscillating circuits is easy and precise, the hundreds of bit patterns require thousands of clock cycles.

The second PUF (see Figure 12) is implemented with an arbiter circuit. The arbiter has two inputs, which are low initially. The arbiter waits for one of the inputs to go high; its output indicates which input went high first. To harden the PUF design to confront potential adaptive modeling attacks, a feedforward arbiter is placed at an intermediate point in the circuit. Its output drives one of the

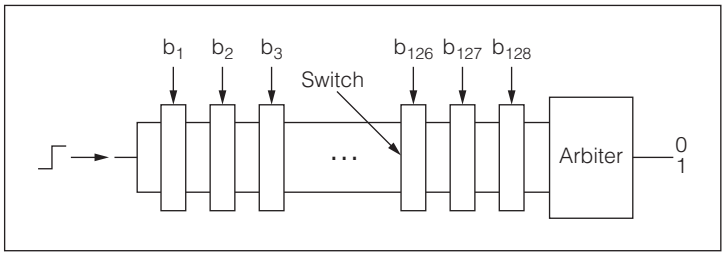

Figure 12. PUF design 2. (Source: Gassend et al.[30])

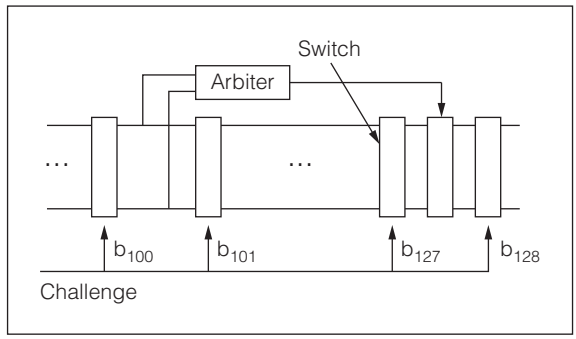

Figure 13. Hardening a PUF via arbiters. (Source: Gassend et al.[30])

switches later in the circuit (see Figure 13). Many such arbiters could be added to the circuit to make modeling by an adversary more difficult. A *variable delay buffer* is a component that directly passes a signal from its input to its output at a different speed, depending on the state of its other input. Thus, the extra effort put into the buffer's variable delay characteristics can have the benefit of making the circuit difficult to model in the additive delay model.

However, recent studies have shown that the feedforward-arbiter approach is susceptible to nonlinear attacks, such as an evolutionary strategy.[31] The method proposed by Majzoobi, Koushanfar, and Potkonjak safeguards the PUFs by using an input network that avoids individual controlling of the challenges, and an XOR output network that compresses the responses so that they cannot be reverse-engineered.[32] Even though attempts have been made to break this type of safeguard, no polynomial method for learning the PUF with an XOR output network has been reported thus far.

To ensure attack resiliency, Majzoobi, Koushanfar, and Potkonjak proposed five tests: predictability, collision, sensitivity, reverse engineering, and emulation.[32] The *predictability test* identifies the level of difficulty of correctly calculating or predicting PUF or random-ID outputs for a given input. The *collision test* studies whether two different inputs can map to the same output. The *sensitivity test* captures the part of the required manufacturing variability that allows a PUF to operate securely when the components are imperfect. The *reverse-engineering test* measures the level of difficulty to find the characteristics of the circuit components. The *emulation test* measures the compressibility level of the PUF I/O pairs. If a PUF is reverse-engineered, the emulation test always requires at most a linear number of operations with respect to the number of participating elements in the PUF.

Majzoobi, Koushanfar, and Potkonjak also introduced a suite of new techniques for testing the security of random IDs, such as true random-number generators (TRNGs), and PUFs.[33] Introducing such techniques is critical for many security applications that use random IDs and PUFs as security primitives or for key generation. The goal is not only to create a methodology for testing the security of random-ID generation and PUFs at the functional and IC levels, but also to define specific techniques and tools that can be extended to other security hardware.

The authors focused on five broad classes of attacks: predictability, collision, fault injection, reverse engineering, and emulation. After presenting an extensive case study of the test method on popular and widely advocated PUF structures, they analyzed two delay-based structures: linear (parallel) and feedforward. They demonstrated that both structures can be reverse-engineered and easily emulated using the introduced tests. They also introduced a set of safeguarding techniques to create secure PUFs. The goal is to foster a clearer understanding of the PUF security requirements, and to propose a testing methodology to evaluate the requirements. They also devised a specific PUF structure that can pass the proposed security tests. The quantitative tests indicate drastically higher levels of achieved security with respect to conventional linear and feedforward PUFs.

So far, our focus has been on trust issues and Trojan detection methods for ASICs and general-purpose microprocessors. Trimberger suggested that the FPGA trust and Trojan insertion problems are markedly different from those of ASICs.[34] This is because the FPGA design process is almost completely separate from the manufacturing flow, disallowing tampering or piracy at fabrication. FPGAs fabricated during nonsecure manufacturing can be trusted if the bitstream is developed and loaded in a secure design facility so that programming secrets are not exposed to manufacturers. Trimberger argued that, because the FPGA bitstream is an electronic message, the known methods of message authentication and piracy are valid. The only issue is that the physical access to the device could enable new attacks. Trimberger also summarized the known methods for securing FPGA systems in the field, including methods that prevent reverse engineering of the bitstream, load-once programming,

Table 1. Summary of recent work on Trojan detection.

Paper	Test modality	Trojan model	Golden model	Detection method	Process variation	Benchmark
Agrawal et al.[13]	Transient power	Counter: 16-bit Comparators: 3, 8-bit	Invasive characterization	Kullback-Leibler (KL) distance		256-bit Rivest, Shamir, and Adleman (RSA)
Wang et al.[9]	Transient power	Counter: 1, 3, 7, 9-bit Comparators: 3, 5, 20-bit	Simulation	Current integration	✓	S38417
Rad et al.[10]	Transient power	Comparator	Simulation	Outlier analysis	✓	C432
Li and Lach[16]	Delay	Inverter chain	Simulation	Path delay analysis using shadow registers		Braun multiplier
Jin and Makris[17]	Delay	2, 4-bit comparator	Simulation	Path delay analysis measurement	✓	Data Encryption Standard (DES) core
Wolff et al.[6]	Functional	2-input gates activated by Trojans	Simulation	Frequency analysis for rate triggers		ISCAS
Banga and Hsiao[19,20]	Transient power, pattern generation	Varying flip-flop numbers	Simulation	Pattern generation for increasing transient power in a Trojan	✓	ISCAS
Potkonjak et al.[15]	Static power, delay	1-gate	Simulation	Pattern generation for increasing static power and circuits' transient switching	✓	ISCAS
Banga and Hsiao[26]	Design for hardware trust, transient power	2, 3, 4-input gates	Simulation	Pattern generation to improve full activation and transient-power analysis	✓	ISCAS
Alkabani and Koushanfar[14]	Static power	1, 3, 5 added 2-input gates	Simulation	Pattern generation to increase leakage transitions, robust statistics, and optimization	✓	ISCAS

and bitstream encryption. According to Trimberger, the serious vulnerability at the foundry is its susceptibility to attacks on the security features, which an adversary could later try to use in the field (e.g., via Trojan or spyware insertion).[34]

Current trends in semiconductor design and manufacturing toward globalization and horizontal integration of the industry constitute a source of threat and vulnerability. Table 1 summarizes some of the key Trojan detection methods discussed in this article. In the table, "test modality" (second column) refers to the measurement modality used (often as a side channel) to reveal the presence of a Trojan. "Trojan model" (third column) indicates the type of added Trojan used in the experimental

designs—for instance, the number of components and their characteristics. "Golden model" (fourth column) represents the environment used to present the results. Except for the work by Agrawal et al.,[13] where the authors had access to test chips and did noninvasive measurements, all methods were based on noninvasive simulations. "Detection method" (fifth column) briefly describes the essence of the detection method of choice in the work cited. "Process variation" (sixth column) identifies whether the Trojan detection algorithms and protocols take into account the inherent variability in the silicon structure. Finally, in the last column, we identify the benchmarks used.

Hardware Trojan modeling and analysis is a growing research topic that has attracted considerable attention during the past three years. The recent work in this area has paved the way for more comprehensive models, preventive strategies, analysis, and detection methods for Trojans. ■

Acknowledgments

We thank Yousra Alkabani, Mehrdad Majzoobi, Hassan Salmani, Michel Wang, and Xhehui Zhang from Rice University and the University of Connecticut, Storrs, for their contributions to this article.

■ References

1. "Report of the Defense Science Board Task Force on High Performance Microchip Supply," Defense Science Board, US DoD, Feb. 2005; http://www.acq.osd.mil/dsb/reports/2005-02-HPMS_Report_Final.pdf.
2. J. Lieberman, "National Security Aspects of the Global Migration of the U.S. Semiconductor Industry," white paper, Airland Subcommittee, US Senate Armed Services Committee, June 2003; http://lieberman.senate.gov/documents/whitepapers/semiconductor.pdf.
3. S. Adee, "The Hunt for the Kill Switch," *IEEE Spectrum,* vol. 45, no. 5, 2008, pp. 34-39.
4. "Innovation at Risk—Intellectual Property Challenges and Opportunities," white paper, Semiconductor Equipment and Materials International, June 2008.
5. X. Wang, M. Tehranipoor, and J. Plusquellic, "Detecting Malicious Inclusions in Secure Hardware: Challenges and Solutions," *Proc. IEEE Int'l Workshop Hardware-Oriented Security and Trust* (HOST 08), IEEE CS Press, 2008, pp. 15-19.
6. F. Wolff et al., "Towards Trojan Free Trusted ICs: Problem Analysis and Detection Scheme," *Proc. Design, Automation and Test in Europe Conf.* (DATE 08), ACM Press, 2008, pp. 1362-1365.
7. Y. Alkabani and F. Koushanfar, "Extended Abstract: Designer's Hardware Trojan Horse," *Proc. IEEE Int'l Workshop Hardware-Oriented Security and Trust* (HOST 08), IEEE CS Press, 2008, pp. 82-83.
8. Y. Jin, N. Kupp, and Y. Makris, "Experiences in Hardware Trojan Design and Implementation," *Proc. IEEE Int'l Workshop Hardware-Oriented Security and Trust* (HOST 09), IEEE CS Press, 2009, pp. 50-57.
9. X. Wang et al., "Hardware Trojan Detection and Isolation Using Current Integration and Localized Current Analysis," *Proc. IEEE Int'l Symp. Defect and Fault Tolerance of VLSI Systems* (DFT 08), IEEE CS Press, 2008, pp. 87-95.
10. R. Rad et al., "Power Supply Signal Calibration Techniques for Improving Detection Resolution to Hardware Trojans," *Proc. IEEE/ACM Int'l Conf. Computer-Aided Design* (ICCAD 08), IEEE CS Press, 2008, pp. 632-639.
11. S. King et al., "Designing and Implementing Malicious Hardware," *Proc. 1st USENIX Workshop Large-Scale Exploits and Emergent Threats* (LEET 08), Usenix Assoc., 2008, pp. 1-8.
12. Y. Alkabani and F. Koushanfar, "Active Hardware Metering for Intellectual Property Protection and Security," *Proc. 16th USENIX Security Symp.,* Usenix Assoc., 2007, pp. 291-306.
13. D. Agrawal et al., "Trojan Detection Using IC Fingerprinting," *Proc. IEEE Symp. Security and Privacy* (SP 07), IEEE CS Press, 2007, pp. 296-310.
14. Y. Alkabani and F. Koushanfar, "Consistency-Based Characterization for IC Trojan Detection," *Proc. IEEE/ACM Int'l Conf. Computer-Aided Design* (ICCAD 09), IEEE CS Press, 2009.
15. M. Potkonjak et al., "Hardware Trojan Horse Detection Using Gate-Level Characterization," *Proc. 46th Design Automation Conf.* (DAC 09), ACM Press, 2009, pp. 688-693.
16. J. Li and J. Lach, "At-Speed Delay Characterization for IC Authentication and Trojan Horse Detection," *Proc. IEEE Int'l Workshop Hardware-Oriented Security and Trust* (HOST 08), IEEE CS Press, 2008, pp. 8-14.
17. Y. Jin and Y. Makris, "Hardware Trojan Detection Using Path Delay Fingerprint," *Proc. IEEE Int'l Hardware-Oriented Security and Trust* (HOST 08), IEEE CS Press, 2008, pp. 51-57.
18. S. Jha and S.K. Jha, "Randomization Based Probabilistic Approach to Detect Trojan Circuits," *Proc. 11th IEEE High Assurance Systems Engineering Symp.,* IEEE CS Press, 2008, pp. 117-124.
19. M. Banga and M. Hsiao, "A Region Based Approach for the Identification of Hardware Trojans," *Proc. IEEE Int'l*

Workshop Hardware-Oriented Security and Trust (HOST 08), IEEE CS Press, 2008, pp. 40-47.
20. M. Banga and M. Hsiao, "A Novel Sustained Vector Technique for the Detection of Hardware Trojans," *Proc. 22nd Int'l Conf. VLSI Design,* IEEE CS Press, 2009, pp. 327-332.
21. I. Verbauwhede and P. Schaumont, "Design Methods for Security and Trust," *Proc. Design, Automation and Test in Europe Conf.* (DATE 07), EDA Consortium, pp. 672-677.
22. G.E. Suh, D. Deng, and A. Chan, "Hardware Authentication Leveraging Performance Limits in Detailed Simulations and Emulations," *Proc. 46th Design Automation Conf.* (DAC 09), ACM Press, 2009, pp. 682-687.
23. G. Bloom, B. Narahari, and R. Simha, "OS Support for Detecting Trojan Circuit Attacks," *Proc. IEEE Int'l Workshop Hardware-Oriented Security and Trust* (HOST 09), IEEE CS Press, 2009, pp. 100-103.
24. D. McIntyre et al., "Dynamic Evaluation of Hardware Trust," *Proc. IEEE Int'l Workshop Hardware-Oriented Security and Trust* (HOST 09), IEEE CS Press, 2009, pp. 108-111.
25. H. Salmani, M. Tehranipoor, and J. Plusquellic, "New Design Strategy for Improving Hardware Trojan Detection and Reducing Trojan Activation Time," *Proc. IEEE Workshop Hardware-Oriented Security and Trust* (HOST 09), IEEE CS Press, 2009, pp. 66-73.
26. M. Banga and M. Hsiao, "VITAMIN: Voltage Inversion Technique to Ascertain Malicious Insertion in ICs," *Proc. 2nd IEEE Int'l Workshop Hardware-Oriented Security and Trust* (HOST 09), IEEE CS Press, 2009, pp. 104-107.
27. M. Abramovici and P. Bradley, "Integrated Circuit Security: New Threats and Solutions," *Proc. 5th Ann. Workshop Cyber Security and Information Intelligence Research: Cyber Security and Information Challenges and Strategies* (CSIIRW 09), ACM Press, 2009, article 55.
28. R.S. Chakraborty, S. Paul, and S. Bhunia, "On-Demand Transparency for Improving Hardware Trojan Detectability," *Proc. IEEE Int'l Workshop Hardware-Oriented Security and Trust* (HOST 08), IEEE CS Press, 2008, pp. 48-50.
29. G. Qu and M. Potkonjak, *Intellectual Property Protection in VLSI Designs: Theory and Practice,* Kluwer Academic Publishers, 2003.
30. B. Gassend et al., "Identification and Authentication of Integrated Circuits: Research Articles," *Concurrency and Computation: Practice & Experience,* vol. 16, no. 11, 2004, pp. 1077-1098.
31. U. Rührmair, J. Sölter, and F. Sehnke, "On the Foundations of Physical Unclonable Functions," Cryptology ePrint Archive, report 2009/277, 10 June 2009; http://eprint.iacr.org/2009/277.pdf.
32. M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Techniques for Design and Implementation of Secure Reconfigurable PUFs," *ACM Trans. Reconfigurable Technology and Systems,* vol. 2, no. 1, 2009, article 5.
33. M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Testing Techniques for Hardware Security," *Proc. Int'l Test Conf.* (ITC 08), IEEE CS Press, 2008, pp. 1-10.
34. S. Trimberger, "Trusted Design in FPGAs," *Proc. 44th Design Automation Conf.* (DAC 07), ACM Press, 2007, pp. 5-8.

Mohammad Tehranipoor is an assistant professor of electrical and computer engineering at the University of Connecticut, Storrs. His research interests include CAD and test for CMOS VLSI designs and emerging nanoscale devices, DFT, at-speed test, secure design, and IC trust. He has a PhD in electrical engineering from the University of Texas at Dallas. He is a senior member of the IEEE and a member of the ACM and ACM SIGDA.

Farinaz Koushanfar is an assistant professor of electrical and computer engineering and the director of Texas Instruments DSP Leadership at Rice University. Her research interests include hardware trust and IP protection, computer and communication system security, design and analysis of adaptive systems, and emerging technologies. She has a PhD in electrical engineering and computer science from the University of California, Berkeley. She is a senior member of the IEEE and a member of the ACM and the American Association for the Advancement of Science.

■ Direct questions and comments about this article to Mohammad Tehranipoor, Electrical and Computer Engineering Dept., University of Connecticut, 371 Fairfield Way, Unit 2157, Storrs, CT 06269-2157; tehrani@engr.uconn.edu.