INVITED PAPER

Hardware Trojan Attacks: Threat Analysis and Countermeasures

This paper is a survey of the state-of-the-art Trojan attacks, modeling, and countermeasures.

By Swarup Bhunia, *Senior Member IEEE*, Michael S. Hsiao, *Fellow IEEE*, Mainak Banga, *Member IEEE*, and Seetharam Narasimhan

ABSTRACT | Security of a computer system has been traditionally related to the security of the software or the information being processed. The underlying hardware used for information processing has been considered trusted. The emergence of hardware Trojan attacks violates this root of trust. These attacks, in the form of malicious modifications of electronic hardware at different stages of its life cycle, pose major security concerns in the electronics industry. An adversary can mount such an attack with an objective to cause operational failure or to leak secret information from inside a chip—e.g., the key in a cryptographic chip, during field operation. Global economic trend that encourages increased reliance on untrusted entities in the hardware design and fabrication process is rapidly enhancing the vulnerability to such attacks. In this paper, we analyze the threat of hardware Trojan attacks; present attack models, types, and scenarios; discuss different forms of protection approaches, both proactive and reactive; and describe emerging attack modes, defenses, and future research pathways.

KEYWORDS | Hardware intellectual property (IP) trust; hardware obfuscation; hardware Trojan attacks; self-referencing; side-channel analysis; Trojan detection; Trojan taxonomy; Trojan tolerance

Manuscript received October 30, 2013; revised May 20, 2014; accepted June 24, 2014. Date of publication July 15, 2014; date of current version July 18, 2014. This research was supported in part by the National Science Foundation (NSF) under Grants CNS-1054744 and DUE-1245756.

S. Bhunia is with the Case Western Reserve University, Cleveland, OH 44106 USA (e-mail: skb21@case.edu).
M. S. Hsiao is with the Virginia Polytechnic Institute and State University, Blacksburg, VA 24061 USA (e-mail: hsiao@vt.edu).
M. Banga is with Intel Corporation, Folsom, CA 95630 USA (e-mail: mainak.banga@intel.com).
S. Narasimhan is with Intel Corporation, Hillsboro, OR 97124 USA (e-mail: seetharam.narasimhan@intel.com).

Digital Object Identifier: 10.1109/JPROC.2014.2334493

I. THE THREAT

Hardware Trojan attacks have emerged as a major security concern for integrated circuits (ICs) [1]–[5]. These attacks relate to malicious modifications of an IC during design or fabrication in an untrusted design house or foundry, which involve untrusted people, design tools, or components. Such modifications can give rise to undesired functional behavior of an IC, or provide covert channels or backdoor through which sensitive information can be leaked. An adversary is expected to make a Trojan stealthy in nature that evades detection through conventional postmanufacturing test, but manifests during long hours of field operation. For an IC with moderate complexity, the number of possible Trojans can be inordinately large with varying activation mechanisms (referred to as triggers) and effects (referred to as payloads). Fig. 1 shows a simplified block diagram of a hardware Trojan, which causes a malfunction (by modifying signal S to S'), when triggered—i.e., when the activation condition realized by the trigger logic is true. Such malicious inclusions effectively act as "spies or terrorists" on chip and can be extremely powerful, potentially leading to catastrophic consequences in diverse applications [15].

These malicious circuits have been popularly referred to as "hardware Trojans" similar to the software Trojans, which attack the operating system (OS) of a computer. The nomenclature is derived from a mythological incident attributed to the ancient Greeks in the Trojan war, where a wooden horse was gifted to the Trojan army who took it into their city walls without realizing that the enemy (Greek) soldiers were hidden inside the hollow horse. The seemingly trustworthy horse conditionally turned into a powerful and malicious weapon that drastically affected the course of the Trojan war. Similar to its mythical analogy, the two main features of a hardware Trojan are as

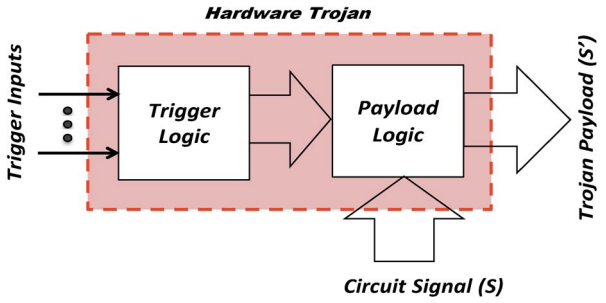

Fig. 1. *General structure of a hardware Trojan in a design.*

follows: 1) it should have a malicious intent; and 2) it should evade detection under conventional postmanufacturing test/validation process.

Current economic trend plays a major role in enhancing the vulnerability to Trojan attacks [1]–[3]. IC design and manufacturing practices increasingly rely on untrusted parties and entities in the IC life cycle. Economic factors dictate that most of the modern ICs are manufactured in unsecured fabrication facilities. Moreover, modern IC design often involves intellectual property (IP) cores supplied by untrusted third-party vendors, outsourced design and test services, as well as electronic design automation (EDA) software tools supplied by different vendors. Such a business model has, to a large extent, relinquished the control that IC design houses had over the design and manufacture of ICs making them vulnerable to malicious implants. Fig. 2 illustrates different steps of a typical IC life cycle and the possibility of Trojan attacks in these steps [3]. Each party associated with the design and fabrication of an IC can be a potential adversary who can tamper it [5]. Such tampering can be accomplished through

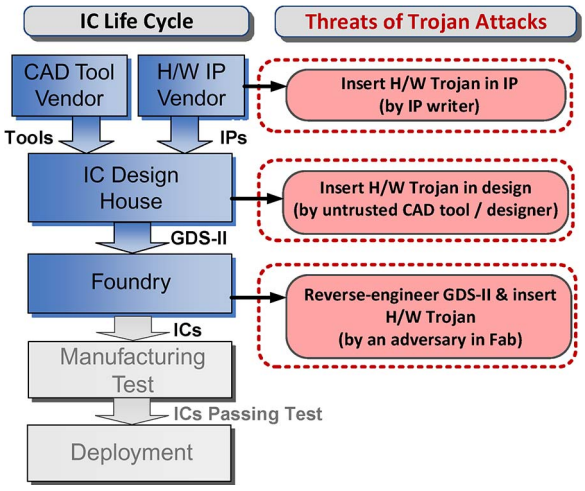

Fig. 2. *Hardware Trojan attacks by different parties at different stages of IC life cycle.*

add/delete/alteration of circuit structure or through modification of manufacturing process steps that causes reliability issues in ICs. From an attacker's perspective, the objective of such attacks can be manifold, e.g., to malign the image of a company to gain competitive edge in the market; disrupt major national infrastructure by causing malfunction in electronics used in mission-critical systems; or leak secret information from inside a chip to illegally access a secure system. Concerns about this vulnerability and the resultant compromise of system security have been expressed globally [3], [10], [11], especially since recent discoveries point to feasibility of such attacks [12]–[14]. Moreover, several unexplained military mishaps in the past have been attributed to the presence of malicious hardware modifications [1], [35]. Recent investigations have shown that an intelligent adversary can mount a hard-to-detect Trojan attack using just a few transistors or logic gates in a large multimillion transistor system-on-chip (SoC) design, or by selectively changing specific process steps, e.g., the doping profile, to affect the operational reliability of a circuit [14].

Ideally, any undesired modification made to an IC should be detectable by pre-silicon verification/simulation or post-silicon testing. However, pre-silicon verification or simulation requires a completely specified golden model of the entire IC. This might not be always available, especially for IP-based designs where IPs can come from third-party vendors. Besides, a large multimodule design is usually not amenable to exhaustive verification [6]. Post-silicon, the design can be verified either through destructive depackaging and reverse engineering of the IC [3], or by comparing its functionality or circuit characteristics with a golden version of the IC [4], [7], [8]. However, state-of-the-art approaches do not allow destructive verification of ICs to be either cost effective or scalable [6], [9]. Moreover, as pointed out in [3], it is possible for the adversary to insert Trojans in only some ICs on a wafer, not the entire population, which limits the usefulness of a destructive approach. Traditional postmanufacturing testing is not suitable for detecting hardware Trojans. This is due to the stealthy nature of hardware Trojans and the vast spectrum of possible Trojan instances an adversary can exploit.

Hardware Trojan attacks raise a new set of challenges for trusted operation of electronics in field [15]. It demands trust validation of ICs with respect to malicious design modification at various stages of the IC lifecycle, where untrusted components/personnel are involved. In particular, it introduces the requirement for reliable detection of any malicious design modification during postmanufacturing test. It also imposes a requirement for trust validation in hardware IP cores obtained from untrusted third-party vendors.

A. Trojan Versus Fault

Postmanufacturing test, both structural and functional, is targeted to detect different types of manufacturing

Table 1 Comparison Between Faults and Hardware Trojan Attacks

	Fault	Hardware Trojan
Activation	Usually at known functional state	Arbitrary combination/sequence of internal circuit states (digital/analog)
Insertion Agent	*Accidental* (due to imperfection in manufacturing process)	*Intentional* (inserted by an adversary during IC design or fabrication)
Manifestation	Functional/parametric failure	Functional/parametric failure or information leakage

defects. Faults such as stuck-at-faults or path delay faults are logical models of physical defects (e.g., resistive short or open). Table 1 compares the properties of hardware Trojans with those of faults. Unlike faults in a design, which occur due to imperfections introduced during the manufacturing process, hardware Trojans are deliberately inserted by an adversary to serve a specific malicious purpose. While a fault is activated at a known functional state of a circuit (e.g., a stuck-at-0 fault at the output of an inverter can be activated by sensitizing its input to a logical value of 0), a Trojan can be designed to activate at an arbitrary complex condition, including a sequence of events at internal circuit nodes.

B. Software Versus Hardware Trojans

Table 2 distinguishes hardware Trojans from its software counterparts in terms of key properties. A software Trojan horse (STH) is a type of malware program with malicious code that gains privileged access to the OS and may steal information or cause harm to the host computer (e.g., erase or corrupt data) [16], [67]. An STH attack can often be cured in the field by running an anti-Trojan program (such as the one in [76]) that monitors and removes the Trojan [68], [69]. Hardware Trojans are inserted into an IC before it is fabricated. Unlike an STH, a hardware Trojan is virtually impossible to remove from a chip after fabrication and hence can be extremely difficult to remedy during field operation.

C. Trojan Attacks Through Hardware IP or CAD Tools

SoC design based on reusable hardware IP is now a pervasive practice in the semiconductor industry due to the dramatic reduction in design/verification cost and time it offers. This growing reliance on reusable preverified hardware IPs and a wide array of design automation tools during SoC design—often gathered from untrusted third-party vendors—severely affects the security and trustworthiness of SoC computing platforms [17]. The possibility of Trojan attacks in third-party IP poses a major integrity concern to SoC designers. Verification of trust of an IP acquired from untrusted third-party sources can be extremely challenging due to lack of golden models. Conventional verification approaches, which rely on the presence of a golden or reference design, do not work in case of third-party IPs. On the other hand, functional simulation or emulation does not provide adequate coverage due to often incomplete functional specifications. Thus, even though simulation can validate the correctness of a design with respect to functional specifications, it cannot provide assurance against additional functionality due to a Trojan. For example, if a processor IP core triggers a malicious memory write for an "add" instruction with a specific rare combination of operand values, both simulation and emulation are very likely to fail, since they may not excite all rare conditions. Similarly, untrusted computer-aided design (CAD) tools can potentially cause malicious inclusions in a design [18]. Such attacks can be introduced early in the design cycle, so that they can evade subsequent verification steps. Often, a designer uses a CAD tool suite from the same vendor. In that case, a verification tool from a vendor can potentially ignore a malicious insertion by the synthesis engine from the same vendor.

D. Complex Attack Mode

Malicious collusion between multiple parties at different stages of the design, manufacturing, and deployment can make a hardware Trojan attack more potent. One such example is presented in [19], where the inserted Trojan circuitry leaks information through a covert side channel that allows conspiring malicious parties to discover the encryption key. Another example is presented in [20], where focused ion beams help insert a dormant Trojan to a

Table 2 Software Versus Hardware Trojan Attacks

	Software Trojan	Hardware Trojan
Activation	A type of malware that resides in a code and activates during its execution	Resides in hardware (e.g. IC) and activates during its operation
Infection	Spreads through user interaction e.g. downloading and running a file from Internet	Inserted through untrusted entities in design or fabrication house
Remedy	Can be *removed in field* through S/W support	*Cannot be removed* once IC is fabricated

functional unit in the fabrication facility. A general model of such an attack, referred to as multilevel attack, is described in [21]. Using a crypto module as a case study, it analytically shows that the resultant attack poses a significantly stronger threat than that from a single adversary. Such attacks can easily bypass both pre-silicon verification as well as postmanufacturing test. An example would be a fault attack in an advanced encryption system (AES) module, which enables only the malicious parties, who are part of the conspiracy, to retrieve the cipher key. The Trojan could also be a passive hardware entity, which simply serves to help malicious software circumvent the hardware protection mechanisms. Moreover, a Trojan circuit can be localized or distributed in a chip, and a trigger condition or the payload of a Trojan can be digital or analog, e.g., a Trojan can be triggered by changes in temperature [15]. Becker *et al.* [60] have demonstrated that by selectively changing the dopant level in the transistors, they are able to create Trojans that are resistant against existing detection techniques. These Trojans are nonintrusive and difficult to detect since they do not alter any functionality and are not optically observable.

II. TROJAN MODEL, INSTANCES, AND CLASSIFICATION

A. Trojan Models and Examples

An intelligent adversary is expected to hide such tampering with an IC's behavior in a way that makes it extremely difficult to detect with conventional postmanufacturing testing [5]. Intuitively, it means that the adversary would ensure that such tampering is manifested or triggered under very rare conditions at the internal nodes, which are unlikely to arise during testing but can occur during long hours of field operation [22], [23]. Fig. 3

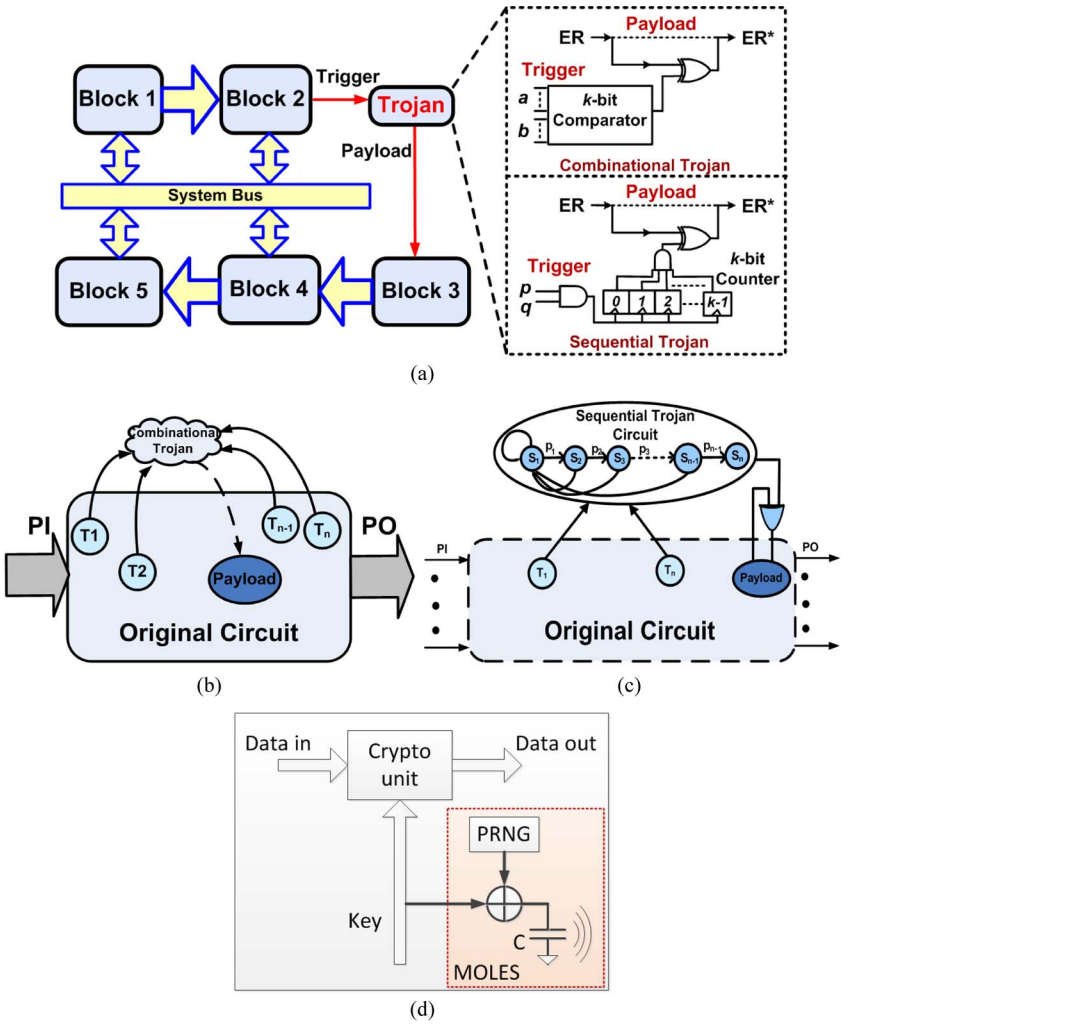

Fig. 3. *Hardware Trojan attacks in different forms: (a) combinational and sequential Trojans; general model of (b) combinational and (c) sequential Trojan, which can facilitate test generation for Trojan detection; (d) Trojans with capability of leaking secret information from inside a crypto chip through power side channels [19].*

shows some examples of different types of hardware Trojans. The combinational Trojan, as shown in Fig. 3(a), does not contain any state element (e.g., flip-flop or latch) and depends only on the simultaneous occurrence of a set of rare node conditions (e.g., a predefined value on node sets a and b) to trigger a malfunction (by flipping signal ER). The sequential Trojan shown in Fig. 3(a), on the other hand, undergoes a sequence of state transitions before triggering a malfunction. Fig. 3(a) shows a synchronous k-bit counter, which activates when the count reaches $2^k - 1$, by modifying the node ER to an incorrect value at node ER^*. Fig. 3(b) and (c) show general models of combinational and sequential Trojans, respectively. These abstract models of Trojans are useful for studying the space of possible Trojans, and, similar to fault models, help in test vector generation for Trojan detection.

1) Trojans in Cryptographic Engines: A possible Trojan attack in a crypto engine can try to subvert the security mechanisms. The payload could range from a mechanism that presents dummy keys, predefined by the attacker, instead of the actual cryptographic keys used for sensitive encryption or signature verification operations, to leaking the secret hardware keys via covert side channels, e.g., information leaked through a power trace. Fig. 3(d) provides an example of such a Trojan that attempts to leak a secret key from inside a cryptographic IC through power side-channels using a technique called malicious off-chip leakage enabled by side channels (MOLES) [19]. Even if the IC has been designed to minimize side-channel information leakage, a hardware modification could help overcome the protection under specific circumstances where the attacker is in possession of the system or physically near the system to extract the secret information. Other targets could be a random number generator used for deriving random session keys for a particular operation or the debug passwords used for unlocking test-mode access to security-sensitive signals. Researchers have also proposed leaking such secret information over wireless channels [77] by using low-bandwidth modulation of the transmitted signal.

2) Trojans in General-Purpose Processors: In general-purpose processors, an attacker at the fabrication facility can implement a backdoor, which can be exploited in the field by a software adversary [23]–[26]. For example, modern processors implement a hardware chain of trust to ensure that malware cannot compromise the hardware assets such as secret keys and memory range protections. By using different stages of firmware and boot code authentication, one can ensure that the operating system (OS) kernel and lower levels (such as hypervisor) are not corrupted. However, in such systems, the attacker at an untrusted fabrication facility could implement a backdoor which disables the secure booting mechanism under certain rare conditions or when presented with a unique rare input condition in the hands of an end-user adversary [23]. Similarly, other objectives which could be realized with the help of hardware Trojans would be to bypass memory range protections using buffer overflow attacks or to gain access to privileged assets by evading the access control protection mechanisms implemented in the hardware.

B. Trojan Taxonomy

The taxonomy of Trojan circuits has been presented in various forms, and it continues to evolve as newer attacks and Trojan types are discovered. Here, we will present a high-level classification, as shown in Fig. 4, based on variations in activation mechanism and Trojan effect. Based on the trigger condition, the hardware Trojans can be classified into analog and digital Trojans. The former are activated by analog conditions such as temperature, delay, or device aging effect, whereas the latter are triggered by some Boolean logic function. Digitally triggered Trojans can again be classified

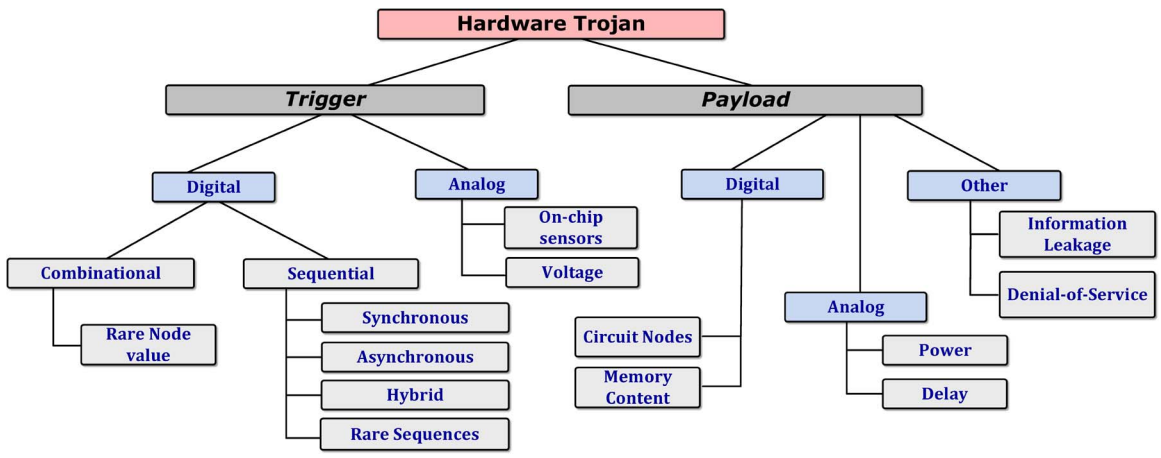

Fig. 4. *Trojan taxonomy based on trigger and payload mechanisms.*

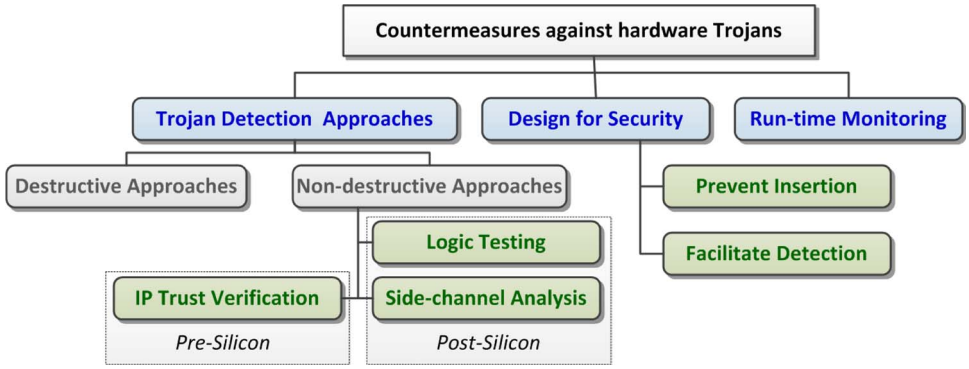

Fig. 5. ***Overview of different protection approaches against hardware Trojan attacks.***

into combinational and sequential types. Analog Trojans include attacks on process steps that compromise reliability of all or select chips. These Trojans, referred to as reliability trojans by some researchers, may cause accelerated aging of the devices [27]. The reduction in reliability is caused by acceleration of the wearing out mechanisms for complementary metal–oxide–semiconductor (CMOS) transistors, such as negative bias temperature instability (NBTI) or hot carrier injection (HCI). Selective malicious changes in the manufacturing process, such as variation in nitrate concentration in the gate oxide layer or the temperature used during the nitrate layering process [27] can result in creation of infected ICs with a much shorter lifetime. What makes it challenging to detect such tampering is that it can be local to a particular functional circuit block or even to a small section of a block that can potentially evade standard postfabrication reliability characterization steps.

In terms of the payload, the Trojan can cause functional failure upon triggering or have a passive effect such as heating of the die or leaking of information. A Trojan can cause an "information leakage" attack, where secret information is leaked by a Trojan via a transmitted radio signal or serial data port interface such as the RS-232-C port. It could also involve a side-channel attack where the information is leaked through the power trace [19] or through thermal radiation or through optical modulation of an output light-emitting diode (LED) [28]. Another type of Trojan payload would be unauthorized alteration in system behavior, e.g., a denial-of-service (DoS) attack, which causes a system's functionality to be unavailable.

Other parameters used to define and compare different Trojans include the hardware overhead and activation probability [5]. The area and power overhead relative to the design in which the Trojan is inserted has to be an undetectable fraction, in order to evade detection by obvious means [29]. One can reuse existing logic and unused states in existing finite state machines (FSMs) to reduce the overhead. Moreover, the layout of complex ICs typically contains unused space which can be used for inserting extra gates without affecting the die footprint. The activation probability is also a tradeoff between avoiding detection and malicious impact.

III. COUNTERMEASURES

A. Challenges

Conventional postmanufacturing tests using functional/structural/random patterns perform poorly to reliably detect hardware Trojans. This is because manufacturing test generation and application attempt to detect defects or unacceptable variations in device parameters that cause deviation from functional or parametric specifications. They do not identify additional functionalities due to a Trojan or deviation in circuit behavior triggered by arbitrary rare events. There are several important challenges with respect to reliable detection of Trojans using a post-silicon test/validation process. First, an adversary can exploit inordinately large number of Trojan instances of varying forms and sizes [30]. These Trojans vary widely in structural and functional properties including trigger conditions and payloads, which makes it difficult to develop a logical model of Trojans. The number of possible Trojan instances has a combinatorial dependence on the number of circuit nodes. As an example, even with the assumption of maximum of four trigger nodes and a single payload, a relatively small ISCAS-85 benchmark circuit $c880$ with 451 gates can have $\sim 10^9$ triggers and $\sim 10^{11}$ distinct Trojan instances, respectively [8]. Thus, it is not practical to enumerate all possible Trojan instances to generate test patterns or compute test coverage. Second, due to their stealthy nature, activating arbitrary Trojan instances and observing their effects can be extremely challenging. Hence, deterministic and exhaustive testing approaches appear infeasible. Third, with respect to observing Trojan effects in physical parameters, e.g., delay and supply current, process and measurement noise pose a major challenge. In particular, increasing process variations in advanced technologies can mask minuscule Trojan effects in a physical parameter.

B. Class of Protections

Research efforts on protection against Trojan attacks have focused on three broad classes of solutions: 1) Trojan detection approaches; 2) design for security (DFS) approaches; and 3) runtime monitoring approaches. Fig. 5 shows a broad classification of the countermeasures against hardware Trojans. Trojan detection approaches typically attempt to detect the existence of Trojans at an IP level using pre-silicon techniques or using nondestructive techniques during post-silicon manufacturing test through a trust validation process. They can be further classified into logic testing approaches, which focus on generating appropriate test patterns for Trojan detection, and side-channel analysis approaches where a measurable parameter such as power, delay, temperature, or electromagnetic (EM) radiation profile can be used to isolate a Trojan-infected IC from the golden ones. The DFS approaches attempt to make insertion of hard-to-detect Trojans difficult or facilitate detection during post-silicon validation. However, the DFS or trust validation approaches usually are not capable of providing complete confidence against diverse forms of possible Trojan attacks. To safeguard against potentially undetected Trojans, runtime validation approaches can be employed based on online monitoring of circuit operation [15], [31]. Such approaches provide a last line of defense against Trojan attacks and often attempt to contain the effect of an activated Trojan (e.g., by entering a fail-safe mode).

C. Why Destructive Reverse Engineering Is Not Effective

Once the fabrication and packaging of an IC is complete, we have limited visibility into the components inside it. One can, however, use destructive reverse-engineering techniques to depackage an IC and obtain microscopic images of each layer to reconstruct the design for trust validation of the end product. The destructive techniques [9], [32] use a sample of the manufactured ICs which are subject to demetallization using chemical mechanical polishing (CMP) followed by scanning electron microscope (SEM) image reconstruction and analysis [3]. Though it would take several weeks to months to do this for an IC of reasonable complexity, it has the potential of giving 100% assurance that any malicious modification in the IC will be detected. However, at the end of this invasive process, the IC cannot be used, and we only get the information for a single IC. It is possible for the attacker to have infected only some samples in an entire lot of ICs by manipulating the layout masks. Hence, in general, destructive approaches are not considered viable for Trojan detection. However, destructive reverse engineering on a limited number of samples can be attractive in order to obtain the characteristics of a golden batch of ICs. This information can be useful for trust validation through side-channel analysis, as discussed in Section IV.

IV. TROJAN DETECTION

Table 3 provides a comparison of advantages and disadvantages of two major Trojan detection paradigms: logic testing and side-channel analysis. Logic testing approaches, both functional and structural, attempt to develop directed test patterns to activate unknown Trojan instances and propagating their effects to output ports [7]. Although robust under process and measurement noise, these approaches are likely to fail to activate large Trojans consisting of large numbers of trigger inputs. An alternative approach is to measure a side-channel parameter, such as supply current or path delay, which can be affected due to unintended design modifications. However, the effectiveness of side-channel analysis is limited by large intrinsic device parameter variations in modern nanometer technologies. These detection approaches typically require a golden design or a set of golden ICs to compare the measured values in order to identify the Trojan-infected ones. Table 4 provides a comparison of the Trojan detection capability of alternative approaches. Logic testing and side-channel-analysis-based validation approaches provide complementary capabilities in detecting Trojans of different types and sizes. Hence, a postmanufacturing validation solution that combines the benefits of both approaches can be effective in maximizing the level of confidence. For applications, which require the highest level of trust against Trojan attack, one can combine postmanufacturing validation with online monitoring. Finally, validation approaches, both postmanufacturing and online, can be complemented with low-cost DFS solutions, which harden a design with respect to Trojan insertion or help in the validation process.

A. Logic Testing

In order to detect the existence of a Trojan using logic testing, it is not only important to satisfy the trigger

Table 3 Comparison of Logic Testing and Side-Channel-Analysis-Based Trojan Detection

	Logic Testing	Side-Channel Analysis
Pros	• Robust under process noise • Effective for detecting ultra-small Trojans	• Effective for large Trojans • Easy to generate test vectors
Cons	• Difficult to generate test vectors • Large Trojan detection challenging	• Vulnerable to process noise • Ultra-small Trojan detection challenging

Table 4 Capability of Trojan Detection Schemes to Identify Different Hardware Trojan Types

Trojan Size / Trojan Type		Large		Small	
		Localized	Distributed	Localized	Distributed
Digital	Combinational	Logic Testing	Side-Channel (IDDT) + Online	Logic Testing + Online	Side-Channel (IDDT)
	Sequential	Side-Channel	Side-Channel (IDDQ)	Side-Channel (IDDT)	Side-Channel (IDDT)
Analog		Side-Channel (IDDT, IDDQ, Others) + Online			

condition, but also to propagate the effect of such an event to an output node and observe it. Due to the inordinately large space of possible Trojans, as mentioned earlier, it is not practical to enumerate all possible Trojan instances to generate deterministic test patterns or compute test coverage. It indicates that, instead of an exact approach, a statistical approach for test vector generation can be computationally more tractable. This has motivated efforts to develop a statistical approach for Trojan detection. One such effort, referred to as multiple excitation of rare cccurrence (MERO) is reported in [8]. The main objective of this methodology is to derive a set of test patterns that is compact (minimizing test time and cost), while maximizing the Trojan detection coverage. The basic concept is to detect low probability conditions at the internal nodes and then derive an optimal set of vectors than can trigger each of the selected low probability nodes individually to their rare logic values multiple times (e.g., at least N times, where N is a user-defined parameter). By increasing the toggling rate of nodes that are random-pattern resistant, it improves the probability of activating a Trojan compared to purely random patterns.

Fig. 6 illustrates the concept with two examples—one combinational and one sequential Trojan. The combinational Trojan in Fig. 6(a) is activated when $a = 0$, $b = 1$, and $c = 1$ are satisfied, while the sequential Trojan in Fig. 6(b) requires occurrences of $a = 1$ and $b = 0$ in several clock cycles to trigger. Hence, if we can generate a set of test vectors that induce these conditions at these nodes individually N times where N is sufficiently large (e.g., > 100), then a Trojan with a triggering condition composed jointly of these nodes is highly likely to be activated by the application of this test set. The MERO methodology is conceptually similar to the N-detect test used in stuck-at automatic test pattern generation (ATPG), where the test set is generated to detect each single stuck-at-fault in a circuit by at least N different patterns in order to improve test quality and defect coverage. Another approach of logic testing is to develop guided tests for detecting Trojan attacks in small but critical parts of a design [33]. For example, one can generate test vectors to observe undesired writes into a memory array or unjustifiable activity in the key-dependent logic of a crypto system. Waksman *et al.* [64] have used the stealthiness property of Trojan-affected nets to isolate them from a given design. Their tool, called FANCI, uses a scalable, Boolean functional analysis to detect these nets.

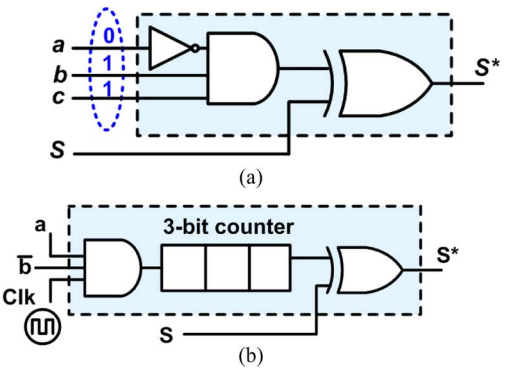

Fig. 6. *Logic testing approach for hardware Trojan detection based on multiple excitation of rare conditions [8]: (a) for combinational Trojan; (b) for sequential Trojan.*

1) Coverage Metric: A metric similar to fault coverage, that provides a measure of confidence against arbitrary Trojan attacks, can be attractive for quantifying the effectiveness of a Trojan detection approach. Due to difficulties in enumerating all possible Trojan instances, it is infeasible to deterministically measure Trojan coverage. Statistical measures of confidence have been developed to address this issue. A random sampling approach can be used to compute both Trigger and Trojan coverage [8]. From the Trojan population, a representative set of Trojans with specific structure (e.g., number of trigger nodes) is randomly selected. From this set of sampled Trojans, ones with false trigger conditions that cannot be justified with any input pattern are eliminated. Then, a circuit under test is simulated for each vector in the given vector set and checked whether the triggering condition is satisfied. For an activated Trojan, if its effect can be observed at any primary output or scan flip-flop input, the Trojan is considered detected. The percentages of Trojans activated and detected constitute the trigger coverage and the Trojan coverage, respectively. To make such measures effective, however, it is important to consider an adequate number

of Trojans in the sample (as dictated by the principles of sampling theory) from the universal Trojan space.

B. Side-Channel Analysis

In order to overcome the limitations of logic testing approaches, the use of side-channel analysis during post-silicon testing has been widely investigated [30], [34]–[42]. A malicious inclusion during design or fabrication is bound to have an effect on the power consumed by the circuit. In most cases, it is also likely to affect the delay of certain circuit paths. The impact on power is expected to be due to addition/alteration of circuit elements used in realizing the Trojan. It can cause deviation in the static (i.e., current drawn at idle state of a circuit) as well as in the transient (i.e., switching) current profile of the circuit under test. Most Trojans constantly monitor their activation condition. The act of monitoring consumes power, even if the change in power consumption is minimal; indeed that would be the goal of an adversary [4]. A delay impact can be due to either addition of extra logic level in a path (as in Fig. 6) or increase in capacitive load in a path. The impact of an unknown Trojan in physical parameters can be observed and compared between golden and Trojan-infected circuits to identify its existence.

Effectiveness of Trojan detection through side-channel analysis largely depends on two major parameters: 1) signal-to-noise ratio (SNR); and 2) the Trojan-to-circuit ratio (TCR). The signal of interest is the effect of a Trojan on a side-channel parameter. Noise is introduced by process and environmental variations. The effect of a Trojan on a side-channel parameter such as current or delay can be easily masked by the noise. Hence, simple comparison techniques can lead to a large number of false detections. This has led to a wide variety of postprocessing techniques to statistically isolate the Trojan effect from noise. On the other hand, the TCR indicates the gap between the original and Trojan-affected parameter. In an ideal scenario of zero noise, the detection sensitivity depends on the TCR. Hence, amplifying Trojan effect compared to the original value helps improving the confidence in identifying a Trojan instance. Zhang *et al.* [62] have proposed a virtual probe framework to get an accurate profiling of spatial variations within the chip without incurring much overhead cost. The problem is formulated as a maximum *a posteriori* (MAP) estimation problem and linear programming is used to solve it. Next, we present several side-channel analysis approaches based on different parameters such as transient current, static current, and path delay, and discuss their challenges and effectiveness.

1) Static Current Analysis: The presence of a Trojan circuit will be reflected in the current drawn from the power supply, even if no switching occurs in the Trojan circuit. Static CMOS gates are subject to leakage current in the idle mode, and this is more pronounced for submicrometer technologies. Any extra gate will consume extra leakage power, which is additive and can ideally be used to distinguish golden circuits from ones with Trojan [34]–[36]. However, the leakage current effect due to a few extra gates often provides poor sensitivity when measuring the total leakage current for a multimillion gate SoC due to low TCR. Because of the exponential dependence of leakage current on transistor threshold voltage (V_T) variations, chip-to-chip variations in leakage current can be very high, leading to low SNR. To increase sensitivity, one can measure current from multiple power pins, as shown in Fig. 7, thus effectively reducing the problem to that of detecting a few gates in a fraction of the total gates in the IC [34]. Since the leakage current of a logic gate is sensitive to the value at its inputs, a region-based approach using appropriate input vectors can also help localize a small region of the IC, which is infected by the Trojan.

Wei and Potkonjak [61] have employed a segmentation-based diagnosis approach to isolate Trojans in affected circuits. They use overall leakage current of the Trojan gates to identify them, even when they are not activated. The circuit under test is partitioned into a number of segments and the segments that dissipate higher leakage current are more closely inspected.

2) Transient Current Analysis: The goal of transient current (I_{DDT}) analysis is to detect switching activity inside a Trojan circuit [30]. A Trojan detection approach based on this mechanism needs to carefully consider natural variations in current flow and minimize its influence. Fig. 8(a) illustrates device parameter variations, both chip to chip (interdie) and within chip (intradie), in a nanometer process. As observed from Fig. 8(a), variations in a device parameter such as V_T manifest themselves in variations at circuit level and mask the effect of Trojan in current and operating frequency ($F_{\max}$). Depending on the threshold, we can trade off the number of false positives

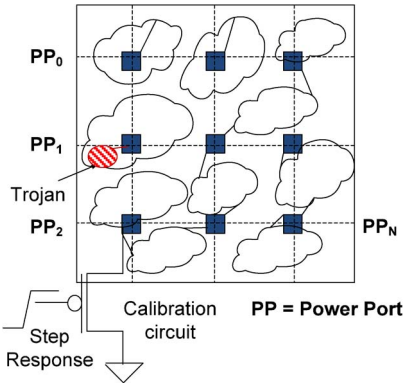

Fig. 7. ***Region-based approach using current measurements from multiple power ports can be used for isolating Trojan effect with high sensitivity [40].***

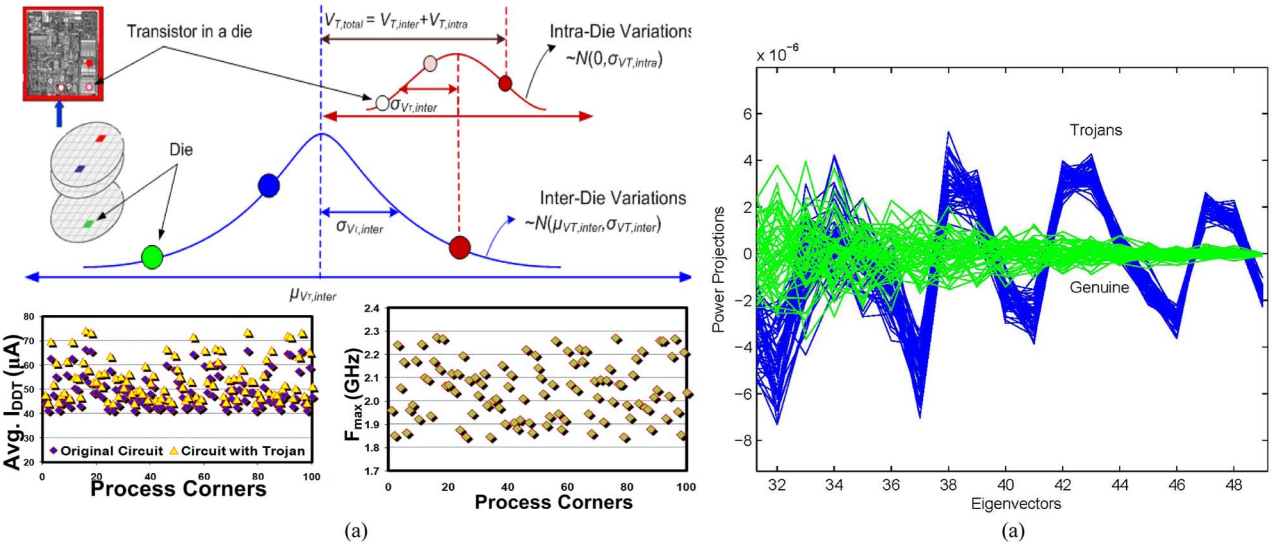

Fig. 8. ***(a) Die-to-die and within-die variation in a device parameter (threshold voltage or V_t) and corresponding impact in side-channel parameters: transient supply current (I_{DDT}) and maximum frequency (F_{max}) [30]. (b) Side-channel transient current signal from a Trojan can be separated and identified from the statistical distribution of process noise using Karhunen–Loeve expansion [4].***

(where we misclassify golden chips as Trojan-containing ICs) and false negatives (where we fail to identify Trojan ICs because they fall under the process noise margin). Similar variations can arise due to temporal effects such as voltage and temperature variations and measurement noise during testing.

The variability in the measurement process can be minimized by standardizing the measurement setup and averaging over multiple measurements to cancel out random noise. The choice of input vector set plays a significant role in improving TCR, thus improving detection sensitivity. Region-based partitioning [43] and directed test vector generation [33] to induce switching activity in possible Trojan instances can cause them to be easily detected. On the other hand, the sensitivity can be increased by decreasing the effect of process noise. Most of the existing approaches try to achieve that through process calibration by normalization and using statistical averaging techniques [37]–[40].

The IC fingerprinting technique [4] uses signal processing techniques such as the Karhunen–Loeve expansion to calibrate the process noise and identify subspaces from the transformed power trace signal, which highlight the presence of the Trojan, as shown in Fig. 8(b). From the analysis of power traces, it is possible to identify reasonably small Trojan instances, e.g., one with an equivalent area of 0.01% of the total size of the circuit, in the presence of ±7.5% random parameter variations. Region-based approaches have been explored where careful selection of test vectors causes switching activity in a smaller functional region of the chip, thus decreasing the denominator of TCR equation. Such a divide-and-conquer method can also take advantage of the different power pins of the chip having more sensitivity to different structural regions of a chip. By calibrating the power profile of these different pins, the effect of process noise can be reduced while highlighting the Trojan contribution to transient current. The interdie process variations can also be neutralized by comparing transient currents from different regions in a chip using a self-referencing matrix [42]. It also helps to localize a Trojan by pinpointing which region of an IC is infected with Trojan circuit. Such a diagnosis can be useful for tracing the source of a Trojan attack.

3) Path Delay Analysis: Another parameter of interest for Trojan detection is the path delay [38], [39]. Activating a Trojan may sensitize a functional path whose propagation delay is adversely affected by the malicious inclusion. However, impact of a Trojan in path delay can be small. In particular, for large path delays, a minor change in delay is likely to be masked by process variations. Hence, appropriate vector selection that leads to sensitization of paths affected by Trojans is of great importance. For sequential circuits without full scan, one needs to adopt design-time techniques for measuring all path delays, including short paths.

To isolate the Trojan effect from a side-channel parameter, the Trojan detection problem can be formulated as a linear programming problem (LPP) with process variation of each gate represented as a constant scaling factor for its leakage or delay under various input conditions [36]. This technique is capable of detecting a single extra gate in benchmark circuits. It can be further

refined with the effect of thermal variations on the leakage to break all correlations. Statistical measures are used to improve the detection accuracy and obtain a high degree of separation between golden and Trojan-affected instances. Like most other side-channel analysis approaches, however, the scalability of such an approach to large designs and its effectiveness in actual measurements require further study.

4) Multiple-Parameter Analysis: In order to reliably isolate the Trojan effect in the presence of process noise, one can use multiple side-channel parameters such as I_{DDT} and F_{max} together to make the Trojan effect more prominent [30]. The measured F_{max} values can be used to calibrate the interdie process corner of a chip. Any variation in I_{DDT}, which does not follow the expected trend due to process variations, may indicate presence of a Trojan. The detection sensitivity can be further improved through proper test vector selection or by using another parameter such as static current. One can also formulate it as a multimodal Trojan detection solution [41] to systematically consider the different parameters and increase the Trojan detection sensitivity.

5) Test Generation for Trojan Detection: Trojans are hard to detect using conventional testing mechanisms, since 100% scan-based fault coverage or full functional coverage cannot guarantee full manifestation of hidden Trojans. In software, methods that use a digital signature to validate the authenticity of the software before running it on the machine are popular [43]. However, at a chip level, there are no efficient notions of static signatures that can detect malicious intrusions. Hence, there is a need for test pattern generation techniques that can effectively detect Trojans of various forms and sizes. Next, we describe two vector generation techniques that exploit the properties of Trojans to detect them with high confidence.

The sustained vector technique [44] tries to drastically reduce the original circuit activity so that any incremental activity in the triggered portion of the Trojan can be highlighted. However, one also needs to ensure that there is at least some kind of activity going on inside the circuit. In other words, the circuit should not be allowed to enter some sleep mode, which can make the Trojan dormant. On the other hand, in the statistical test generation approach, the circuit is first partitioned into smaller subcircuits, which are called regions [45]. The radius defines the extent of a region. Fig. 9 illustrates the concept of region and radius in a circuit. For a gate, the region around it comprises all the transitive fan-in and fan-out gates that are within the defined radius. Thus, a single gate constitutes a region of radius zero, immediate fan-in and fan-out gates along with the original gate constitutes a region of radius one, and so on. The regions are restricted across clock boundaries, i.e., no gates crossing flip-flops are included in a region. Once the regions have been defined, the technique attempts to create a minimal test set that maximizes the activity on a per-region basis.

6) The Need for a Golden Model: Side-channel-analysis-based Trojan detection typically depends on the existence of golden models or golden chips. Simulation-based analysis is typically not considered effective for process calibration due to potential inaccuracies in the simulation model and inherent uncertainties in the manufacturing process. The set of golden chips can be obtained in two ways: 1) by destructive reverse engineering to ensure that they are trusted; and 2) through exhaustive testing to verify their trustworthiness. Both processes are highly expensive and time consuming. Furthermore, it may not be practically feasible to comprehensively verify trustworthiness of even few chips through these processes. This has motivated researchers to develop side-channel-analysis-based countermeasures that do not require golden models. One such approach, called temporal self-referencing (TeSR), eliminates the requirement of golden chip instances by comparing a chip's transient current signature with itself, but at a different time window [72]. When a Trojan-free

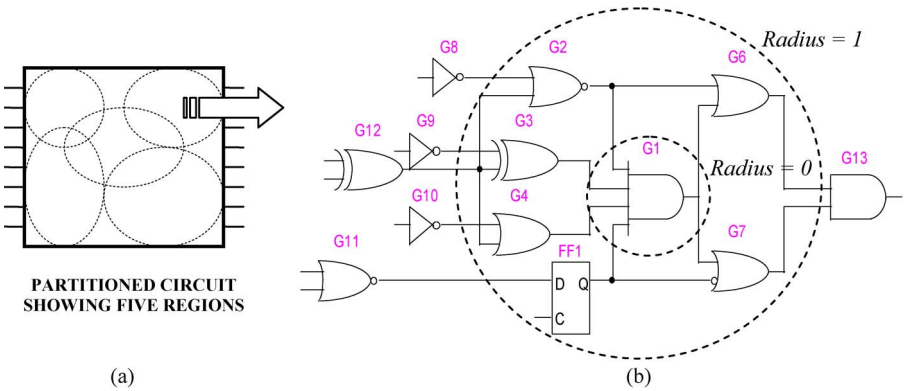

Fig. 9. *Illustration of the concept of: (a) "region," and (b) "radius" in a circuit during test vector generation [45].*

circuit is made to undergo the same set of state transitions multiple times, the transient current "signature" should remain constant over different time windows. However, in a Trojan-infected circuit, the current signature varies over multiple time windows due to uncorrelated switching activity in the Trojan circuit. Such a temporal self-referencing approach can provide high detection sensitivity for Trojans of varying sizes. TeSR, however, applies only to sequential Trojans, and its effectiveness may be limited by the measurement noise. Structural self-similarity in a design can also be exploited to accomplish golden-free Trojan detection. The basic idea is to compare side-channel signature (e.g., current, delay) from a block (e.g., a full adder) with that from another similar block on the same chip. Natural symmetry in path delays in a circuit block can be used to efficiently isolate a Trojan-infected path from another by comparing delay signature of similar paths [73]. These self-referencing approaches rely on the fact that the Trojan effect is local and not all similar components are affected by a Trojan. In addition to not requiring a golden model, they automatically eliminate the effect of interdie process noise in a side-channel parameter, thus improving detection sensitivity [72].

C. Trojan Detection in Hardware IPs

Today's SoC designs use a large number of IP cores from different IP vendors, with varying degree of reliability associated with each vendor. Trustworthiness of these third-party IPs is imperative to ensure trustworthiness of the SoCs, in which they are used. Trojans may be inserted into IPs of different forms by a rogue designer or an untrusted CAD tool in an IP design house. Existing solutions for trusted IP acquisition fall into three broad classes. The first one relies on trust verification of IPs through directed test and verification. Conventional wisdom dictates modeling the IP trust verification problem as a formal verification problem. However, lack of a reliable golden design makes direct application of such verification approaches infeasible. One can, however, employ suspect-signal-guided sequential equivalence checking (SEC) [47] to gain some degree of confidence on the netlist. In the first step, functional vectors are generated using sequential ATPG to drop unsuspicious signals and identify suspect candidates. In the next step, *N*-detect full-scan ATPG is used to further discard the signals, which correspond to combinational untestable stuck-at-faults, by allowing the system to reach even those states, which are not functionally reachable. Then, the suspect circuit and specification-derived circuit are compared using FSM unrolling to see if the same behavior is produced when the suspect signal is activated. The approach, however, heavily relies on proper selection of the suspect signals. In [46], a structural checking approach is suggested to verify integrity of IPs, but the technique is not easily scalable to large designs. Recent works have also proposed using multiple copies of the same IP from different vendors to compare using unrolling [65] and validation [66] techniques.

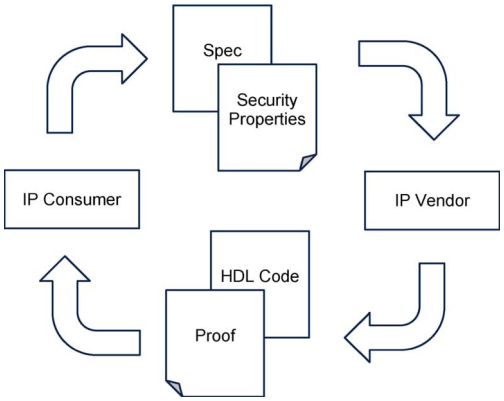

Fig. 10. *IP acquisition and delivery protocol based on PCC [17].*

They can identify malicious inclusions assuming that different vendors will not introduce identical Trojan instances.

The second class of approaches relies on a design paradigm for facilitating the acquisition of trustworthy IP. One such framework is based on proof-carrying code (PCC) [17]. Fig. 10 shows the protocol of trusted IP transfer between an IP writer and user. A set of security-related properties is formulated, and a formal proof of these properties is crafted by the designer. Any undesired modification to the IP is likely to violate the proofs. Validation of security-related properties is carried out by the IP user using the PCC to ensure that no hardware description language (HDL) code was modified or tampered with. Although the security-related properties cannot ensure complete trust in an IP vendor who crafted the formal proof of these properties, it nevertheless offers a line of defense against malicious alteration.

A new paradigm of expert-system-based IP analysis approaches [70], [71] forms the third class of solutions for IP trust verification. These approaches capture prior and trusted experience about designs, which include their specifications, into one or more knowledge bases (KBs). Next, KB rules are used to perform static/dynamic analysis to check correspondence to specifications. Primary challenge in these approaches lies in building the KBs. The knowledge for a standard IP can be captured by building specification ontologies, property-based models, and rule-base and associated assertion libraries. The manual effort is initially high and minimal afterwards. However, the reuse of KB across multiple IC design projects justifies the initial effort.

V. DESIGN FOR SECURITY

Side-channel analysis and test generation for Trojan activation are widely discussed methods to detect Trojans.

These methods, though promising, must deal with major challenges due to rare activating nets in the circuit, process variations, and measurement noise. To improve the effectiveness of these detection methods, ICs must be designed with these detection strategies in mind. In fact, trust must be considered as an important design criterion in the design flow of modern ICs instead of being an afterthought.

A. Design Approaches to Prevent Trojan Insertion

Preventive approaches can be broadly classified into two major categories: 1) obfuscation-based approaches; and 2) layout-filler approaches. The first class of approaches is based on obscuring functional and structural properties of a design, thereby making it difficult for an attacker to incorporate Trojans [48], [49]. The central idea is to employ a key-based obfuscation technique that modifies state transition function of a given circuit. It makes a circuit operate in two distinct modes—the normal mode and the obfuscated mode [49]. While normal mode of operation results in desired output, obfuscated mode produces incorrect functional behavior for some input patterns. Such a modification obfuscates the rareness of the internal circuit nodes, thus making it difficult for an adversary to insert hard-to-detect Trojans. Fig. 11 illustrates the modification in the state transition diagram. It can also make some inserted Trojans benign since they can be active only in the obfuscated mode. The combined effect leads to higher Trojan detectability and higher level of protection against such attack. The approach, however, incurs hardware overhead and cannot provide high coverage against random Trojan insertion. Additionally, it requires application of a key to enable normal operation, which imposes the requirement for on-chip storage of key or disabling of the key by the designer after manufacturing.

The second class of approaches aims at denying opportunity for insertion of additional circuit components in a design by filling vacant spaces. To prevent an attacker from identifying filled spaces in a layout and replace them with Trojan circuit, it is important to hide them appropriately. Built-in-self-repair (BISA) is a technique to fill all the unused spaces in an IC with functional standard cells instead of filler cells. Such an approach makes it challenging for an adversary to find any real estate on the device to insert Trojans [50]. BISA is a self-authenticating system, i.e., any attempt for tampering or changing the BISA architecture would be easy to detect. The flow involves identifying the vacant spaces on the design layout, placing the BISA cells to fill in the gaps, routing, and connecting these cells in such a way that they are self-authenticating after the fabrication. BISA provides protection against removal attacks where one or more of the filler standard cells are removed to place the Trojan cells by producing an incorrect signature during self-authentication. It also protects against redesign attacks and resizing attacks, using the same signature difference. Filler approaches including BISA, however, cannot prevent malicious alteration of a set of transistors or addition of a circuit that do not require extra layout space. Moreover, a resourceful adversary can redesign specific part of a circuit (e.g., resizing some transistors or constrained logic synthesis) to create room for Trojan insertion.

An emerging approach to prevent Trojan attack relies on a split-manufacturing process [51]. In this case, the front-end-of-the-line (FEOL) and bank-end-of-the-line (BEOL) process steps are performed in separate fabrication facilities to hide the design intent and prevent malicious insertion. It considers fabrication of device layers (i.e., the FEOL steps) using an advanced process in an untrusted fabrication facility and high-level interconnects (i.e., the BEOL steps) to be processed using a less advanced trusted fabrication facilities. By not sharing the interconnection information to FEOL facilities, it prevents an attacker from understanding a design as well as from incorporating any undesired alteration since it requires connecting a set of transistors, e.g., realizing a Trojan circuit in Fig. 3(a), through a BEOL process.

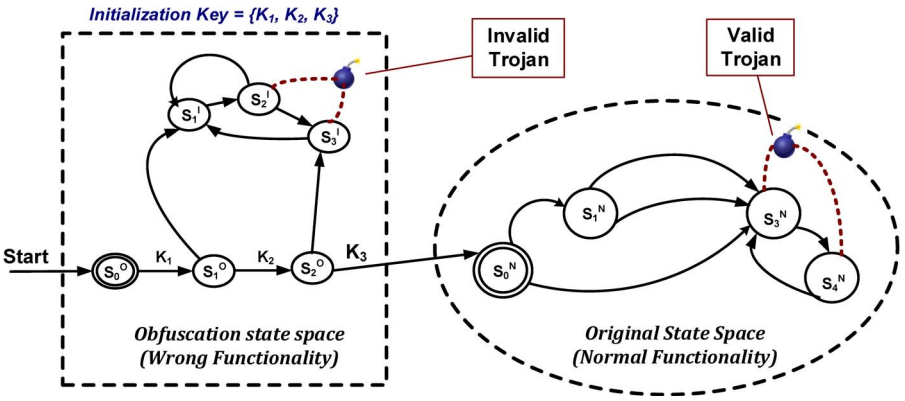

Fig. 11. *Hardware-obfuscation-based design solutions hide design information to thwart Trojan attacks [49].*

B. Design Approaches to Facilitate Trojan Detection

Similar to testing for faults and functional bugs, Trojan detection can also benefit from specially crafted on-chip embedded structures. It needs to address the challenges due to rare activating nets in the circuit, process variations, and measurement noise [52]–[58]. Some of the major DFS approaches that aim at addressing them are described next.

1) On-Chip Security Monitors: The major challenge with side-channel analysis is that a small Trojan (e.g., of size 1–100 transistors) in a large multimillion transistor design would induce a barely noticeable effect in delay or supply current, even with the best available measuring instrument and judicious vector generation. In other words, the "detection sensitivity," which is measured as the percentage deviation in delay or current due to a Trojan, is too low to identify a Trojan reliably. The problem of reduced detection sensitivity is greatly accentuated by the noise due to process and environmental fluctuations [52]. Researchers have proposed configuring circuit paths into ring oscillators [53], [54] to precisely detect small delay variations due to a Trojan. Moreover, integration of transient current sensors in a chip is shown to provide significantly higher detection sensitivity than conventional off-chip current monitoring [52]. Moreover, they provide a scalable solution for designs of arbitrary size and complexity as well as Trojans of various forms and sizes. One can exploit existing power-gating circuits used in modern designs to reduce the overhead of current sensors. Fig. 12(a) illustrates insertion of transient current monitors (CMs) in a chip. The detection sensitivity can improve significantly under process variations, as shown in Fig. 12(b). These on-chip structures, however, are themselves vulnerable to tampering by an adversary in an untrusted fabrication facility. Hence, they require additional validation step to ensure their integrity. In [63], Waksman and Sethumadhavan have proposed two area-efficient detection mechanisms, called TRUSTNET and DATAWATCH, to detect attacks on microprocessors by implanting intelligent insiders within the chip.

2) Removing Rare-Triggered Nets: The stealthy nature of Trojans suggests that they connect to nets with low controllability and/or observability. As an example, a Trojan can have $q > 1$ trigger inputs which can be nets with 1) very low transition probabilities; and 2) rare combinations. When the transition probability of Net_i is very low, either $P_i(0) \gg P_i(1)$ or $P_i(1) \gg P_i(0)$. With q number of trigger inputs, the probability of generating a specific trigger vector is $P_{\mathrm{trigger-vector}} = \prod P_i$ $(i = 1 \text{ to } q)$, assuming independence of the nets. It is expected that $P_{\mathrm{trigger-vector}}$ is very low if P_is are low. By increasing the transition probability of nets with low transition rate, it is possible to eliminate hard-to-activate sites in a design. Scan architecture allows access to internal cells of the circuits, thereby improving controllability and observability for internal nodes. To remove hard-to-activate sites, dummy scan flip-flops can be inserted to increase transition probability of design nets with transition probability less than a threshold (P_{th}) [56]. Such design modifications can be done even if the gate-level netlist is not trusted.

3) Increasing TCR: While efficient vector generation can aim at achieving the goal of increasing TCR, as described earlier, one can achieve the same effect with appropriate design modification. It is possible to reorder scan cells based on their final physical location in the layout. Layout-aware scan-cell reordering can localize switching activity to one region while limiting it in other regions in a design

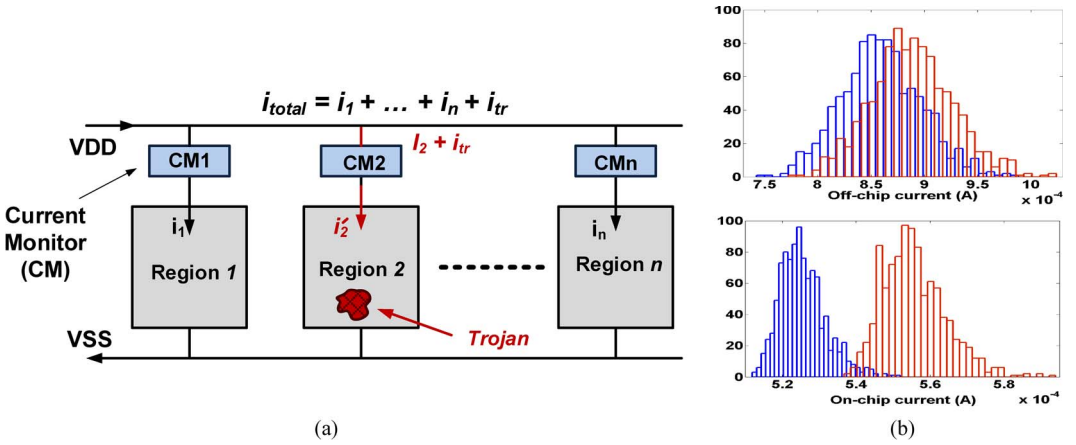

Fig. 12. *(a) On-chip current monitors attached to power supply bumps can significantly improve the detection sensitivity in current-based side-channel analysis. (b) Comparison of off-chip versus on-chip current without (blue) and with Trojan (red) considering effect of process variations [52].*

[55]. Simulation results show that this technique is very effective for small and large Trojans as well as distributed and localized Trojans. This is because reduction in circuit switching using this technique is substantially larger than the reduction in Trojan's switching, even if an attacker distributes the Trojan gates among different regions in the circuit. Contribution of Trojan to the total supply current can also be enhanced through voltage inversion [58]. Let us consider a four-input Trojan AND gate. When we invert the supply voltage of a CMOS gate, it behaves as its complement. Hence, once the voltage is inverted, the Trojan gate becomes a NAND gate and, for random input patterns, the triggering scenario occurs more frequently than in the normal (noninverted) operation. The approach is, however, effective for specific forms of Trojans, and suffers from scalability to large circuits.

VI. RUNTIME MONITORING

Although detecting Trojans before ICs are deployed in the field is highly desirable, the existing techniques cannot guarantee comprehensive coverage of all types and sizes of Trojans. Hence, online monitoring of computations can significantly reduce the potential catastrophic effect of Trojan attacks. These approaches can be used to disable the chip upon detection of malicious logic or to bypass it and allow reliable operation, albeit with some performance overhead, in the presence of unreliable components. Next, we describe the major classes of these monitoring approaches.

A. Configurable Security Monitors

One approach for such online monitoring is based on addition of reconfigurable logic in a given SoC to enable real-time functionality monitoring using security monitors (SMs) [6], [15]. Fig. 13 shows a SoC designed with SMs.

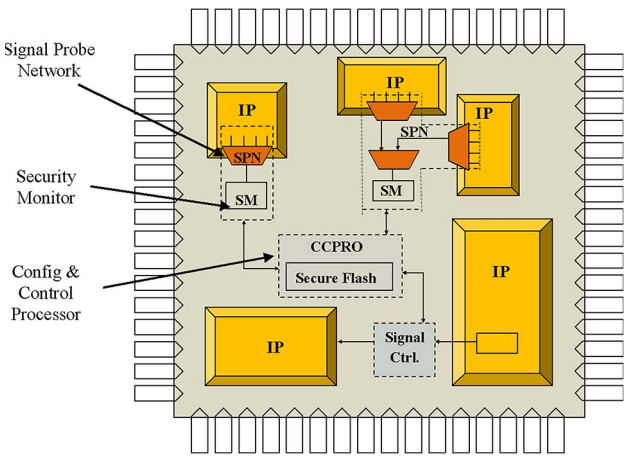

Fig. 13. *Runtime monitoring of Trojan effect using a reconfigurable infrastructure [20].*

SM is configured to implement finite state machines (FSMs) which check the behavior of signals of interest. These signals are fed to SMs through a signal probe network (SPN). The configuration of one SM does not interrupt the normal system operation or other SMs. The checks can be performed concurrently with the normal circuit operation and trigger appropriate countermeasures when a deviation from normal functionality is detected. SMs can be reconfigured by a configuration and control processor to dynamically implement different security checks to detect unexpected or illegal behavior created by a Trojan such as access to protected memory space or entering test mode during normal operation. Typically, the reconfigurable core will implement the functionality of the infected logic, which, in turn, will be disabled or bypassed.

B. Variant-Based Parallel Execution

Another approach combines multicore hardware with dynamic distributed software scheduling to determine hardware trust during in-field use at runtime [31]. It involves scheduling and execution of functionally equivalent variants (obtained by different compilations, or different algorithm variations) simultaneously on different processing elements (PEs) capable of executing these variants and comparing the results. If a mismatch is detected, a new PE is engaged until a match is possible and the PEs with Trojans are identified. The idea is similar to *N*-version programming in software that generates and executes multiple functionally equivalent versions of the same program to achieve high reliability in presence of software faults. Even if two PEs can have similar Trojans, since they execute variants of a code, the Trojans are likely not to be activated simultaneously. It can achieve high level of trust for specific computer systems (e.g., many-core processor) at the cost of additional computation that can lead to performance and energy overhead. Effectiveness of such an approach also relies on efficient generation of variants.

C. Hardware–Software Approach

In case of microprocessor-based systems, a software solution that can detect Trojan activation and/or tolerate Trojan effects can provide effective protection. For example, a simple verifiable "hardware guard" module external to the processor can be used for runtime execution monitoring to identify Trojan activation, as described in [23]. It targets primarily DoS and privilege escalation attacks using periodic checks by the OS, which is enhanced with live check functionality. It can be implemented with only 2.2% average performance overhead based on SPECint 2006 benchmark programs. A hybrid hardware/software approach referred to as *BlueChip* [24] is also investigated which includes a design-time component as well as runtime monitoring. It tries to identify any unused circuitry with design verification tests and tags it as suspicious. At runtime, the suspicious

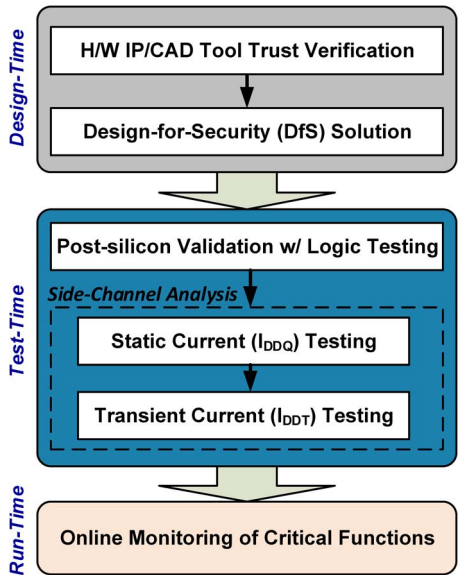

Fig. 14. ***Integrative protection approach that combines the benefits of design, test, and online monitoring solutions and hence can provide the highest level of confidence.***

circuitry is removed and replaced with exception logic, which can trigger a software exception, allowing the system to perform normally by providing a detour around malicious hardware Trojans. This technique is designed to circumvent hardware Trojans, which are similar to software Trojans in purpose. These Trojans can be inserted in a hardware IP or in the instruction code running on an embedded processor.

VII. SUMMARY AND FUTURE DIRECTIONS

The threat of hardware Trojan attacks during IC design and fabrication is escalating with increasing complexity of modern SoCs coupled with the evolving business model. Due to ever-growing computing demands, modern SoCs tend to include many heterogeneous processing cores (e.g., MPSoC), scalable communication network, together with reconfigurable cores, e.g., embedded field-programmable gate array (FPGA), in order to incorporate logic that is likely to change as standards and requirements evolve. Such design practices greatly increase the number of untrusted components in a SoC design. Growing reliance on untrusted third-party IPs, tools, and reduced control on the design/fabrication steps make modern ICs increasingly vulnerable to malicious manipulation. At the same time, new and more complex attack models, such as the ones that take advantage of collusion between IC design and test stages [21], are emerging. With growing awareness among the design as well as manufacturing communities on the possibility of Trojan attacks, adversaries are likely to inflict unanticipated attacks, which are difficult to detect or prevent by existing design and test methods. Attacks such as leakage of confidential information through side channels [19] (e.g., power trace), selectively changing the process parameters, e.g., dopant polarity of transistors [60] during fabrication to compromise lifetime of an IC without affecting functional or parametric behavior, are some of the key examples. The latter does not require any additional logic element or interconnect. These attacks can be extremely stealthy and can bypass even very fine-grain detection techniques.

A "silver bullet" solution, which can reliably protect against Trojan attacks of all types, forms, and sizes is extremely difficult to achieve. On the other hand, an integrative solution, which combines the complementary benefits of design and test solutions, is expected to provide comprehensive coverage. Hence, protection approaches that integrate the benefits of a Trojan-aware design with efficient post-silicon test/validation as well as online monitoring can be attractive for high-assurance applications. Fig. 14 outlines the steps of such a cross-layer integrative protection approach. It includes trust verification of hardware IPs/tools, incorporating design-for-security measures in circuit blocks, postmanufacturing trust evaluation of ICs (through functional and parametric testing), and, finally, runtime monitoring of circuit operation. From the economic standpoint, however, it may be advantageous to invest more in the assurance of individual components (tools and IPs) than protecting the whole IC (through trust verification and monitoring) [59]. Emerging solutions such as nondestructive reverse engineering-based (e.g., through functional and side-channel analysis) trust validation, different forms of design obfuscation, and split-fabrication process [74], [75] are creating new pathways for protecting against these attacks.

The operating principles of emerging nanoscale devices can alter the concepts of hardware Trojan insertion and corresponding defense mechanisms. For example, the current flowing through nanoscale transistors strongly depends on channel stress, which can be modified through changes in process steps or even small layout changes, giving rise to new vulnerabilities. A more interesting problem will be to create reliability Trojans where malicious changes are engineered to radically accelerate device aging. Similar Trojans are also possible in noncharge-based devices, e.g., by marginally modulating spin polarization, or magnetic tunnel junction (MTJ) resistance. On the other hand, the level of intrinsic process variations in both charge- and noncharge-based emerging devices would affect the efficacy of side-channel analysis in detecting Trojan attacks.

Future work would focus on three major areas related to hardware Trojans: 1) exploring emerging attacks, in particular, attacks on circuits made with emerging

nanoscale devices; 2) developing a unified trust metric for coverage estimation, which can be extremely valuable in providing confidence level against arbitrary attacks; and 3) novel protection approaches. Future research on protection approaches would include improving detection sensitivity for unknown Trojans and eliminating the need of golden ICs, Trojan-resilient design, and efficient online detection approaches. ■

REFERENCES

[1] S. Adee, "The hunt for the kill switch," *IEEE Spectrum*, vol. 45, no. 5, pp. 34–39, May 2008.

[2] J. Kumagai, "Chip detectives," *IEEE Spectrum*, vol. 37, no. 11, pp. 43–48, Nov. 2000.

[3] Defense Advanced Research Projects Agency (DARPA), "TRUST in integrated circuits (TIC)," 2007. [Online]. Available: http://www.darpa.mil/MTO/solicitations/baa07-24

[4] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *Proc. IEEE Symp. Security Privacy*, 2007, pp. 296–310.

[5] M. Tehranipoor and F. Koushanfar, "A survey of hardware Trojan taxonomy and detection," *IEEE Design Test Comput.*, vol. 27, no. 1, pp. 10–25, Jan.–Feb. 2010.

[6] M. Abramovici and P. Bradley, "Integrated circuit security—New threats and solutions," in *Proc. 5th Annu. Workshop Cyber Security Inf. Intell. Res.*, 2009, DOI: 10.1145/1558607.1558671.

[7] R. S. Chakraborty, S. Narasimhan, and S. Bhunia, "Hardware Trojan: Threats and emerging solutions," in *Proc. IEEE Int. High Level Design Validation Test Workshop*, 2009, pp. 166–171.

[8] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia, "MERO: A statistical approach for hardware Trojan detection," in *Proc. Workshop Cryptogr. Hardware Embedded Syst.*, 2009, pp. 396–410.

[9] Chipworks Inc. "Semiconductor manufacturing —Reverse engineering of semiconductor components, parts and process.," [Online]. Available: http://www.chipworks.com

[10] Defense Science Board. "Task force on high performance microchip supply," 2005. [Online]. Available: http://www.acq.osd.mil/dsb/reports/ADA435563.pdf

[11] Australian Government, Department of Defence (DoD), Defence Science and Technology Organisation (DSTO). "Towards countering the rise of the silicon Trojan," 2008. [Online]. Available: http://dspace.dsto.defence.gov.au/dspace/bitstream/1947/9736/1/DSTO-TR-2220%20PR.pdf

[12] A. Rawnsley, "Fishy chips: Spies want to hack-proof circuits," *Wired*, Jun. 24, 2011. [Online]. Available: http://www.wired.com/dangerroom/2011/06/chips-oy-spies-want-to-hack-proof-circuits/

[13] R. Johnson, "The Navy bought fake Chinese microchips that could have disarmed U.S. missiles," *Business Insider*, Jun. 27, 2011. [Online]. Available: http://www.businessinsider.com/navy-chinese-microchips-weapons-could-have-been-shut-off-2011-6

[14] S. Skorobogatov and C. Woods, "Breakthrough silicon scanning discovers backdoor in military chip," *Cryptographic Hardware and Embedded Systems Workshop*, vol. 7428, Berlin, Germany: Springer-Verlag, 2012, pp. 23–40.

[15] S. Bhunia *et al.*, "Protection against hardware Trojan attacks: Towards a comprehensive solution," *IEEE Design Test Comput.*, vol. 30, no. 3, pp. 6–17, May–Jun. 2013.

[16] C. E. Landwehr, A. R. Bull, J. P. McDermott, and W. S. Choi, "A taxonomy of computer program security flaws," *ACM Comput. Surv.*, vol. 26, no. 3, pp. 211–254, 1994.

[17] E. Love, Y. Jin, and Y. Makris, "Proof-carrying hardware intellectual property: A pathway to trusted module acquisition," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 1, pp. 25–40, Feb. 2012.

[18] M. Potkonjak, "Synthesis of trustable ICs using untrusted CAD tools," in *Proc. Design Autom. Conf.*, 2010, pp. 633–634.

[19] L. Lin, W. Burleson, and C. Paar, "MOLES: Malicious off-chip leakage enabled by side-channels," in *Proc. Int. Conf. Comput.-Aided Design*, 2009, pp. 117–122.

[20] M. Abramovici, "Protecting integrated circuits from silicon Trojan horses," Military Embedded Syst., Jan.–Feb., 2009. Available: http://www.mil-embedded.com/articles/id/?3748

[21] S. Ali, D. Mukhopadhyay, R. S. Chakraborty, and S. Bhunia, "Multi-level attack: An emerging threat model for cryptographic hardware," in *Proc. Design Autom. Test Eur. Conf. Exhibit.*, 2011, DOI: 10.1109/DATE.2011.5763307.

[22] S. Jha and S. K. Jha, "Randomization based probabilistic approach to detect Trojan circuits," in *Proc. IEEE High Assurance Syst. Eng. Symp.*, 2008, pp. 117–124.

[23] G. Bloom, B. Narahari, and R. Simha, "OS support for detecting Trojan circuit attacks," in *Proc. IEEE Int. Workshop Hardware-Oriented Security Trust*, 2009, pp. 100–103.

[24] M. Hicks, M. Finnicum, S. T. King, M. M. K. Martin, and J. M. Smith, "Overcoming an untrusted computing base: Detecting and removing malicious hardware automatically," in *Proc. IEEE Symp. Security Privacy*, 2010, pp. 159–172.

[25] S. T. King *et al.*, "Designing and implementing malicious hardware," in *Proc. 1st USENIX Workshop Large-Scale Exploits Emergent Threats*, 2008, article 5.

[26] X. Wang, S. Narasimhan, A. Krishna, T. Mal-Sarkar, and S. Bhunia, "Software exploitable hardware Trojan attacks in embedded processor," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst.*, 2012, pp. 55–58.

[27] Y. Shiyanovskii *et al.*, "Process reliability based trojans through NBTI and HCI effects," in *Proc. NASA/ESA Conf. Adaptive Hardware Syst.*, 2010, pp. 215–222.

[28] F. Kiamilev and R. Hoover, "Demonstration of hardware Trojans," presented at the DEFCON 16, Las Vegas, NV, USA, Aug. 8–10, 2008.

[29] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware Trojans," *IEEE Computer*, vol. 43, no. 10, pp. 39–46, Oct. 2010.

[30] S. Narasimhan *et al.*, "Hardware Trojan detection by multiple-parameter side-channel analysis," *IEEE Trans. Comput.*, vol. 62, no. 11, pp. 2183–2195, Nov. 2013.

[31] D. McIntyre, F. Wolff, C. Papachristou, and S. Bhunia, "Dynamic evaluation of hardware trust," in *Proc. IEEE Int. Workshop Hardware-Oriented Security Trust*, 2009, pp. 108–111.

[32] J. A. Kash, J. C. Tsang, and D. R. Knebel, "Method and apparatus for reverse engineering integrated circuits by monitoring optical emission," U.S. Patent 6 496 022 B1, 2002.

[33] M. Banga, M. Chandrasekar, L. Fang, and M. Hsiao, "Guided test generation for isolation and detection of embedded Trojans in ICs," in *Proc. 18th ACM Great Lakes Symp. VLSI*, 2008, pp. 363–366.

[34] J. Aarestad, D. Acharyya, R. Rad, and J. Plusquellic, "Detecting Trojans though leakage current analysis using multiple supply pad IDDQs," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 4, pp. 893–904, Dec. 2010.

[35] Y. Alkabani and F. Koushanfar, "Consistency-based characterization for IC Trojan detection," in *Proc. Int. Conf. Comput.-Aided Design*, 2009, pp. 123–127.

[36] M. Potkonjak, A. Nahapetian, M. Nelson, and T. Massey, "Hardware Trojan horse detection using gate-level characterization," in *Proc. Design Autom. Conf.*, 2009, pp. 688–693.

[37] D. Acharyya and J. Plusquellic, "Calibrating power supply signal measurements for process and probe card variations," in *Proc. IEEE Int. Workshop Current Defect Based Test.*, 2004, pp. 23–30.

[38] D. Rai and J. Lach, "Performance of delay-based Trojan detection techniques under parameter variations," in *Proc. IEEE Int. Workshop Hardware-Oriented Security Trust*, 2009, pp. 58–65.

[39] Y. Jin and Y. Makris, "Hardware Trojan detection using path delay fingerprint," in *Proc. IEEE Int. Workshop Hardware Oriented Trust Security*, 2008, pp. 51–57.

[40] C. Lamech, R. M. Rad, M. Tehranipoor, and J. Plusquellic, "An experimental analysis of power and delay signal-to-noise requirements for detecting Trojans and methods for achieving the required detection sensitivities," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pt. 2, pp. 1170–1179, Sep. 2011.

[41] F. Koushanfar and A. Mirhoseini, "A unified framework for multimodal submodular integrated circuits Trojan detection," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 1, pp. 162–174, Mar. 2011.

[42] D. Du, S. Narasimhan, R. S. Chakraborty, and S. Bhunia, "Self-referencing: A scalable side-channel approach for hardware Trojan detection," in *Proc. 12th Int. Conf. Cryptogr. Hardware Embedded Syst. Workshop*, 2010, pp. 173–187.

[43] M. A. Williams, "Anti-Trojan, Trojan detection with in-kernel digital signature testing of executables," Security Software Engineering: NetXSecure NZ Limited, Tech. Rep., 2002, pp. 1–12.

[44] M. Banga and M. Hsiao, "A novel sustained vector technique for the detection of hardware Trojans," in *Proc. 22nd Int. Conf. VLSI Design*, 2009, pp. 327–332.

[45] M. Banga and M. Hsiao, "A region based approach for the detection of hardware Trojans," in *Proc. IEEE Symp. Hardware-Oriented Security Trust*, 2008, pp. 40–47.

[46] S. Smith and J. Di, "Detecting malicious logic through structural checking," in *Proc. IEEE Region 5 Tech. Conf.*, 2007, pp. 217–222.

[47] M. Banga and M. Hsiao, "Trusted RTL: Trojan detection methodology in pre-silicon designs," in *Proc. IEEE Int. Workshop Hardware-Oriented Trust Security*, 2010, pp. 56–59.

[48] J. Rajendran *et al.*, "Securing processors against insider attacks: A circuit-microarchitecture co-design approach," *IEEE Design Test*, vol. 30, no. 2, pp. 35–44, Mar.–Apr. 2013.

[49] R. S. Chakraborty and S. Bhunia, "Security against hardware Trojan attacks using key-based design obfuscation," *J. Electron. Testing, Theory Appl.*, vol. 27, no. 6, pp. 767–785, Dec. 2011.

[50] K. Xiao and M. Tehranipoor, "BISA: Built-in-self-authentication for preventing hardware Trojan insertion," in *Proc. IEEE Int. Workshop Hardware-Oriented Trust Security*, 2013, pp. 45–50.

[51] F. Imeson, A. Emtenan, S. Garg, and M. V. Tripunitara, "Securing computer hardware using 3D integrated circuit (IC) technology and split manufacturing for obfuscation," in *Proc. 22nd USENIX Conf. Security*, 2013, pp. 495–510.

[52] S. Narasimhan, W. Yueh, X. Wang, S. Mukhopadhyay, and S. Bhunia, "Improving IC security against Trojan attacks through integration of security monitors," *IEEE Design Test Comput.*, vol. 29, no. 5, pp. 37–46, Oct. 2012.

[53] J. Rajendran, V. Jyothi, O. Sinanoglu, and R. Karri, "Design and analysis of ring oscillator based Design-for-trust technique," in *Proc. IEEE 29th VLSI Test Symp.*, 2011, pp. 105–110.

[54] X. Zhang and M. Tehranipoor, "RON: An on-chip ring oscillator network for hardware Trojan detection," in *Proc. Design Test Eur. Conf. Exhibit.*, 2011, DOI: 10.1109/DATE.2011.5763260.

[55] H. Salmani, M. Tehranipoor, and J. Plusquellic, "A layout-aware approach for improving localized switching to detect hardware Trojans in integrated circuits," in *Proc. IEEE Int. Workshop Inf. Forensics Security*, 2010, DOI: 10.1109/WIFS.2010.5711438.

[56] H. Salmani, M. Tehranipoor, and J. Plusquellic, "A novel technique for improving hardware Trojan detection and reducing Trojan activation time," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 1, pp. 112–125, Jan. 2012.

[57] M. Banga and M. Hsiao, "ODETTE: A non-scan design-for-test methodology for Trojan detection in ICs," in *Proc. IEEE Int. Workshop Hardware-Oriented Trust Security*, 2011, pp. 18–23.

[58] M. Banga and M. Hsiao, "VITAMIN: Voltage inversion technique to ascertain malicious insertions in ICs," in *Proc. IEEE Int. Workshop Hardware-Oriented Trust Security*, 2009, pp. 104–107.

[59] J. Grossklags, N. Christin, and J. Chuang, "Secure or insure?: A game-theoretic analysis of information security games," in *Proc. 17th Int. Conf. World Wide Web*, 2008, pp. 209–218.

[60] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson, "Stealthy dopant-level hardware Trojans: Extended version," *J. Cryptogr. Eng.*, vol. 4, no. 1, pp. 1–13, Apr. 2014.

[61] S. Wei and M. Potkonjak, "Scalable hardware Trojan diagnosis," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 6, pp. 1049–1057, Jun. 2012.

[62] W. Zhang *et al.*, "Virtual probe: A statistical framework for low-cost silicon characterization of nanoscale integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 12, pp. 1814–1827, Dec. 2011.

[63] A. Waksman and S. Sethumadhavan, "Tamper evident microprocessors," in *Proc. IEEE Symp. Security Privacy*, 2010, pp. 173–188.

[64] A. Waksman, M. Suozzo, and S. Sethumadhavan, "FANCI: Identification of stealthy malicious logic using Boolean functional analysis," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2013, pp. 697–708.

[65] T. Reece, D. B. Limbrick, and W. H. Robinson, "Design comparison to identify malicious hardware in external intellectual property," in *Proc. IEEE 10th Int. Conf. Trust Security Privacy Comput. Commun.*, Changsha, China, 2011, pp. 639–646.

[66] C. Liu, J. Rajendran, C. Yang, and R. Karri, "Shielding heterogeneous MPSoCs from untrustworthy 3PIPs through security-driven task scheduling," in *Proc. IEEE Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst.*, 2013, pp. 101–106.

[67] C. E. Landwehr, A. R. Bull, J. P. McDermott, and W. S. Choi, "A taxonomy of computer program security flaws," *ACM Comput. Surv.*, vol. 26, no. 3, pp. 211–254, 1994.

[68] L. A. Hughes and G. J. DeLone, "Viruses, worms, and Trojan Horses: Serious crimes, nuisance, or both?" Social Sci. Comput. Rev., vol. 25, no. 1, pp. 78–98, 2007.

[69] G. McGraw and G. Morrisett, "Attacking malicious code: A report to the Infosec Research Council," *IEEE Software*, vol. 17, no. 5, pp. 33–41, Sep.–Oct. 2000.

[70] B. Singh *et al.*, "Knowledge-guided methodology for third-party soft IP analysis," in *Proc. 27th Int. Conf. VLSI Design*, 2014, pp. 246–251.

[71] B. Singh *et al.*, "Cross-correlation of specification and RTL for soft IP analysis," in *Proc. Design Autom. Test Eur. Conf. Exhibit*, 2014, DOI: 10.7873/DATE2014.303.

[72] S. Narasimhan, X. Wang, D. Du, R. S. Chakraborty, and S. Bhunia, "TeSR: A robust temporal self-referencing approach for hardware Trojan detection," in *Proc. IEEE Int. Symp. Hardware-Oriented Security Trust*, 2011, pp. 71–74.

[73] N. Yoshimizu, "Hardware Trojan detection by symmetry breaking in path delays," in *Proc. IEEE Int. Symp. Hardware-Oriented Security Trust*, 2014.

[74] R. W Jarvis and M. G. McIntyre, "Split manufacturing method for advanced semiconductor circuits," U.S. Patent 7 195 931, 2004.

[75] J. V. Rajendran, O. Sinanoglu, and R. Karri, "Is split manufacturing secure?" in *Proc. Design Autom. Test Eur. Conf. Exhibit.*, 2013, pp. 1259–1264.

[76] TrojanHunter help file. [Online]. Available: http://www.trojanhunter.com/trojanhunter/help/

[77] Y. Liu, Y. Jin, and Y. Makris, "Hardware Trojans in wireless cryptographic ICs: Silicon demonstration & detection method evaluation," in *Proc. Int. Conf. Comput.-Aided Design*, 2013, pp. 399–404.

ABOUT THE AUTHORS

Swarup Bhunia (Senior Member, IEEE) received the B.E. degree (honors) from Jadavpur University, Kolkata, India, in 1995, the M.Tech. degree from the Indian Institute of Technology (IIT), Kharagpur, India, in 1998, and the Ph.D. degree from Purdue University, West Lafayette, IN, USA, in 2005.

Currently, he is a T.&A. Schroeder Associate Professor of Electrical Engineering and Computer Science at Case Western Reserve University, Cleveland, OH, USA. He has over ten years of research and development experience with over 150 publications in peer-reviewed journals and premier conferences in the area of very large scale integration (VLSI) design, computer-aided design (CAD), and test techniques. His research interests include low-power and robust design, hardware security and protection, adaptive nanocomputing and novel test methodologies. He has worked in the semiconductor industry on register transfer level (RTL) synthesis, verification, and low-power design for about three years.

Dr. Bhunia received the IBM Faculty Award (2013), the National Science Foundation (NSF) CAREER development award (2011), the Semiconductor Research Corporation (SRC) Inventor Recognition Award (2009), the SRC technical excellence award (2005), and several best paper awards/nominations. He has served as a Guest Editor of IEEE Design & Test of Computers (2010, 2012), Guest Editor of the IEEE Journal on Emerging and Selected Topics in Circuits and Systems (2014), and on the Editorial Board of the *Journal of Low Power Electronics*. He has served as Program Chair for the 2013 IEEE/ACM NANOARCH and the 2014 International Symposium on VLSI Design and Test (VDAT), and will serve as Program Chair for the 2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). He served in the technical program committees of major IEEE and Association for Computing Machinery (ACM) conferences.

Michael S. Hsiao (Fellow, IEEE) received the B.S. degree in computer engineering (highest honors) and the M.S. and Ph.D. degrees in electrical engineering from the University of Illinois at Urbana-Champaign, Urbana, IL, USA, in 1992, 1993, and 1997, respectively.

Currently, he is a Professor in the Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA, USA. His current research interests include design, testing, verification, and trust of hardware and software.

Prof. Hsiao was a recipient of the Digital Equipment Corporation Fellowship, the McDonnell Douglas Scholarship, and the National Science Foundation CAREER Award. Among his publications, he has been recognized for the most influential papers in the first ten years (1998–2007) of Design Automation and Test Conference in Europe (DATE). He received the best paper award at the 2010 IEEE Asian Test Symposium and the best student paper award at the 2012 IEEE International Test Conference.

Prof. Hsiao was an Associate Editor for the IEEE TRANSACTIONS ON COMPUTERS, the *ACM Transactions on Design Automation of Electronic Systems*, and IEEE DESIGN & TEST OF COMPUTERS. He served as Program Chair for the 2014 IEEE Hardware Oriented Security and Trust Symposium, in addition to serving on technical program committees for numerous conferences.

Mainak Banga (Member, IEEE) received the B.E. degree in electronics and communication engineering (Gold Medalist) from Birla Institute of Technology, Mesra, India, in 2003 and the M.S. and Ph.D. degrees in computer engineering from the Virginia Polytechnic Institute and State University, Blacksburg, VA, in 2008 and 2010, respectively.

Currently, he is a Software Engineer with Intel Corporation, Folsom, CA, USA, in the System Validation and Engineering Group. He is involved in developing software tools to aid per- and post-silicon debug. He worked extensively on Trojan detection mechanisms in pre- and post-silicon IC design flows in his graduate research under the guidance of Prof. M. S. Hsiao. His publications have been selected in renowned conferences like Design Automation and Testing in Europe (DATE), Hardware-Oriented Security and Trust (HOST), VLSI Design, Asian Test Symposium, and Great Lake Symposium on VLSI. His research interest also includes creating faster and more efficient methodologies for design for test.

Seetharam Narasimhan received the B.E. (honors) degree in electronics and telecommunication engineering from Jadavpur University, Kolkata, India, in 2006 and the Ph.D. degree in computer engineering from Case Western Reserve University, Cleveland, Ohio, USA, in 2012.

Currently, he is a Component Design Engineer with Intel Corporation, Hillsboro, OR, USA, in the System Validation and Engineering (SVE) Security Center of Excellence. His areas of interest include hardware security evaluation of system-on-chip products. He has worked extensively on hardware Trojan design and detection as part of his graduate research under the guidance of Prof. S. Bhunia. He has two book chapters and more than 30 publications in peer-reviewed journals and premier conferences in the areas of hardware security and biomedical very large scale integration (VLSI) design.

Dr. Narasimhan has served as a peer reviewer for various IEEE conferences and journals and in the Technical Program Committee for several IEEE/ACM conferences such as the Design Automation Conference (DAC), the International Symposium on Quality Electronic Design (ISQED), VLSI Design-India, the Workshop on NanoArchitecture (NANOARCH), and the International Conference on Computer Design (ICCD). He has been a student competition finalist at the IEEE EMBS conference in 2009, finalist at the CSAW Embedded Systems Challenge in 2009–2011, and received the best paper nomination at the 2010 IEEE Symposium on Hardware-Oriented Security and Trust (HOST).