IIPS: Infrastructure IP for Secure SoC Design

Xinmu Wang, *Member, IEEE,* Yu Zheng, *Student Member, IEEE,* Abhishek Basak, *Student Member, IEEE,* and Swarup Bhunia, *Senior Member, IEEE*

Abstract—Security is becoming an increasingly important parameter in current system-on-chip (SoC) design due to diverse hardware security attacks that can affect manufacturers, system designers or end users. To effectively address the security issues, design-time considerations, i.e. incorporation of Design-for-Security (DfS) features, are becoming essential. However, DfS measures for diverse security threats require specific design modifications to achieve target security level, which significantly increases design effort thus time-to-market, and usually incurs considerable design overhead. In addition, the general heterogeneous architecture of current SoCs makes many core level DfS mechanisms unusable at SoC level. In this paper, we propose a centralized on-chip infrastructure IP for SoC security (IIPS), which alleviates the SoC designers from separately addressing different security issues through design modifications in multiple cores. It also provides ease of integration and functional scalability. We consider a specific implementation of IIPS that provides protection against (1) scan-based attack for information leakage through low-overhead authentication; (2) counterfeiting attacks through integration of a Physical Unclonable Function (PUF); and (3) hardware Trojan attacks through a test infrastructure for trust validation. To make the IP amenable for plug-and-play during SoC design, working protocols of the security functions are designed to comply with IEEE 1500 Standard for Embedded Core Test (SECT). Since IIPS resides outside the functional modules, it does not incur functional performance or power overhead. Simulations and experiments on example SoC designs validate the effectiveness of IIPS in providing protections against diverse attacks at a low hardware overhead.

Keywords—System-on-chip Security, Infrastructure IP, Hardware Trojan, PUF, Scan-based Attack

1 Introduction

Security is becoming an increasingly important parameter in modern System-on-Chip (SoC) design. This is primarily due to the fact that for various forms of existing and emerging security attacks at hardware level, post-manufacturing detection alone cannot provide adequate protection. Fig. 1(a) displays diverse hardware security issues at different stages of SoC development and deployment cycle. To effectively protect SoC hardware against these threats, design-time considerations are becoming mandatory as a mechanism to prevent an attack or facilitate detection (or recovery) in the event of an attack. Major threats that are currently being considered in academia and industries include hardware Trojan attacks in the form of malicious modifications of a design; hardware intellectual property theft, such as illegal sale or use of soft IP cores/ICs; and physical attacks on cryptographic systems during deployment, e.g. side-channel attack, fault-based attack, and scan-based attack. Different design solutions have already been reported to protect against these attacks. For hardware Trojan threats, rare-event removal [15] and ring-oscillator (RO) network [13] have been proposed as design-for-security (DfS) approaches to facilitate Trojan detection or partially prevent Trojans from functioning. For hardware IP protection, techniques like Physical Unclonable Function (PUF) [4], IC metering [8] and hardware obfuscation [12] have been proposed to achieve passive or active protection. For power analysis attacks on crypto modules, various countermeasures have been reported which require design modification at architecture [1] or circuit level [2]. Similarly, several secure scan architectures have been presented earlier [5] [11] [6] [7] to prevent scan-based attack. These solutions collectively show that DfS approaches can provide effective countermeasures against different threats.

However, with the emerging trend of IP-based SoC design, incorporating DfS features in a SoC faces the following major challenges: 1) It requires design of each IP to be modified at design time, thus considerably increasing design effort, hardware overhead, and time-to-market; 2) Separate DfS features need to be incorporated to protect against multiple attacks, which can impose conflicting design as well as test requirements; and 3) The heterogeneous architecture of many modern SoCs, makes core-level DfS measures difficult to apply at SoC level. The situation is aggravated by the prevalent use of third-party IP cores during SoC design, which prohibits arbitrary modifications in a IP. Moreover, many of these techniques incur large silicon area and power overhead, and may cause considerable performance degradation. For example [2] increases the area and power consumption of a crypto core by more than 200%, and the one in [3] incurs 38% performance overhead. The DfS scheme presented in [11] causes significant performance degradation and suffer from scalability issue. Therefore,

• *X. Wang, Y. Zheng, A. Basak and S. Bhunia are with the Department of Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, USA, 44122.*
E-mail: {xxw58, yxz402, axb594, skb21}@case.edu

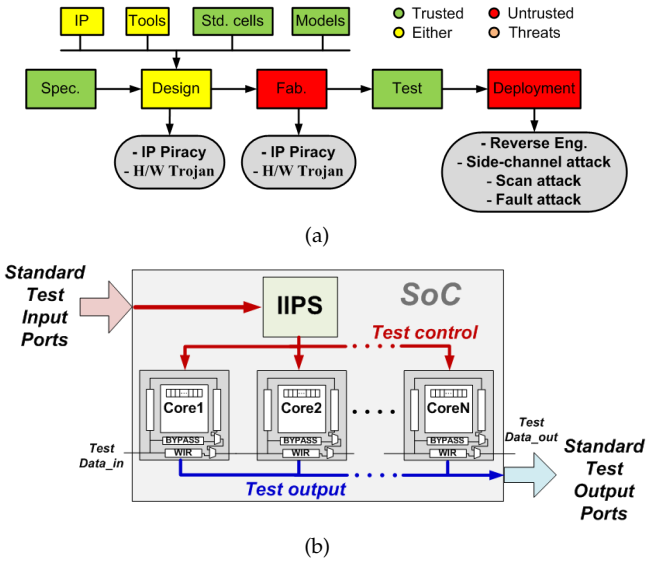

Fig. 1. (a) Security threats at different stages of IC development and deployment cycle; (b) Proposed infrastructure IP for security (IIPS) interfaces with constituent cores of a SoC and provides a flexible, convenient means of addressing various security concerns.

it is desirable to have an easy-to-integrate scalable IP that serves as a centralized resource in a SoC to achieve comprehensive protection against diverse attacks at low design and hardware overhead. Such an IP resembles an infrastructure IP for SoC verification or test [16] [19], but is dedicated to provide security against various attacks.

In this paper, we propose an infrastructure IP for SoC security, referred to as *IIPS*. IIPS can efficiently interface with constituent cores in a SoC through the use of IEEE 1500 Standard Embedded Core Test (SECT), as illustrated in Fig. 1(b). It can provide protection against diverse attacks by either preventing an attack or facilitating detection/recovery during manufacturing test. This centralized module acts as a plug-and-play reusable core for a SoC designer. We have presented general design of IIPS and its interface with other cores. Furthermore, we have considered a specific case study using three major attack models. First, it provides protection against scan-based attack that aims at information leakage through illegal access of a scan chain. The protection is realized through an authentication process, which conditionally activates the scan chains within IPs based on checking a key. Next, it integrates a low-overhead *Physical Unclonable Function (PUF)* primitive for device authentication or cryptographic key generation. It uses a prevalent Design-for-Testability (DfT) structure, namely, the scan chain to implement the PUF. Finally, it implements an infrastructure (based on path delay analysis) for trust validation in presence of hardware Trojan attacks. We show that all the countermeasures exploit the existing test wrappers in IPs to greatly minimize the overhead.

We show that IIPS requires minimal modifications to functional cores for interfacing, thus facilitating system-level integration. Since it resides outside the functional cores and only activates when performing test or security tasks, it does not incur functional performance/power overhead. Our analysis shows that the die-area overhead is very low for realistic SoCs, and is further minimized by sharing the same infrastructure among multiple security primitives. In particular, major contributions of the paper are as follows:

(1) The paper proposes, for the first time to our knowledge, an on-chip infrastructure IP, which can provide protection against multiple security threats for a SoC.

(2) It describes an efficient, general interface of IIPS with functional IP cores through standard SoC boundary scan architecture [25]. It presents appropriate low-overhead design modifications in the IP cores for interfacing with IIPS.

(3) For three dominant threat models, it studies the design of IIPS and its interface. In particular, it considers: (i) scan based attack, (ii) counterfeiting attack, and (iii) hardware Trojan attack. It presents the design requirements for protection against these attacks and optimizes IIPS implementation for hardware overhead.

(4) It presents simulation as well as measurement results to validate the IIPS module functionally, evaluate the hardware overhead and verify its capability in achieving SoC security against the attacks considered.

The remainder of the paper is organized as follows. Section 2 provides background of infrastructure IP and embedded core test standard. Section 3 presents an overview of IIPS, followed by the design details in Section 4. Section 5 describes the SoC level test protocol. Simulation and experimental results are presented in Section 6. Section 7 discusses functional flexibility, scalability and integrity of IIPS. We conclude in Section 8.

2 Background

2.1 Infrastructure IP

Infrastructure IPs (IIPs) refer to a range of IPs that are dedicated to facilitate SoC functional verification, testing or yield improvement. Synopsys Inc. provides a set of verification IPs that can be instantiated in a SoC to verify certain bus protocols [16]. These IPs are targeted to benefit front-end SoC designers, and do not exist on the manufactured SoCs. There are also IIPs that are on-chip modules to be fabricated in order to facilitate post-manufacturing test or debug or to improve yield. Examples include the structures proposed in [19] to help in-field SoC test, for embedded timing characterization [20], transient-error tolerance [21], and for mixed-signal yield improvement [22]. Several vendors provide test-IP products for test capability enhancement for board-level designs [17]. The demand for on-chip dedicated infrastructure logic to facilitate test and debug is rising due to increasing SoC defect rate and test time/cost [18].

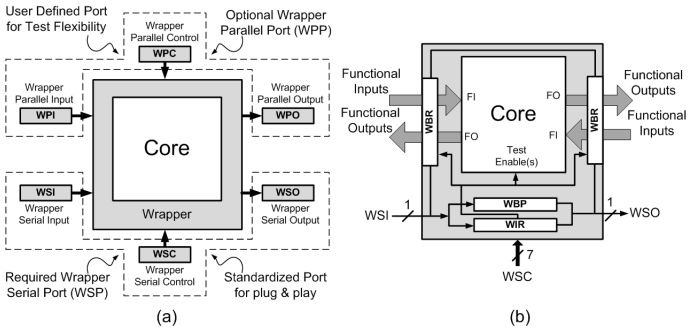

Fig. 2. IEEE 1500 Standard: (a) core wrapper interface terminals; (b) mandatory components of the wrapper [23].

The proposed infrastructure IP for security shares the same motivation and design principles as IIP for test.

2.2 IEEE 1500 Standard

The general heterogeneous architecture of SoCs limits the controllability and observability of internal functional IP cores affecting test coverage. It requires incorporating certain infrastructure logic at design time to allow effective testing of IP cores. Besides, test reuse is becoming an indispensable part in SoC development [23]. It creates a need for a standard test protocol associated with the test infrastructure hardware so that different IP cores can be tested with a unified test framework, and core level test sets created by the core developer can be easily expanded to SoC level. IEEE Std. 1500 is the test standard developed to serve these purposes.

IEEE Std. 1500 [25] contains two parts: (1) a core test wrapper architecture for test access to embedded cores; and (2) a core test language (CTL) [24] for core test knowledge transfer. To comply with the standard, each IP core in the SoC should have a test wrapper. The wrapper provides one boundary register cell for each functional I/O port of the core, where all boundary register cells are referred to as *Wrapper Boundary Register (WBR)*. It also contains a *Wrapper Instruction Register (WIR)* and a *Wrapper Bypass Register (WBR)*. Fig. 2 displays a high-level core wrapper interface and the mandatory components of a core wrapper.

Test access to the embedded cores is enabled by the core wrapper and an on-chip Test Access Mechanism (TAM). Fig. 2(a) shows two ways of accessing the cores: the mandatory *Wrapper Serial Ports* (*WSP*) and the optional *Wrapper Parallel Ports* (*WPP*). WSP provides a single bit test data port (*WSI/WSO*) along with test clock (*WRCK*) and control signals (*WSC*), while WPP offers a larger bit-width for more efficient test execution. However, IEEE Std. 1500 does not anticipate a WPP port, for which the design responsibility resides with the core provider. In this work, IIPS design exploits WSP. However, it can be extended to also use WPP, if available.

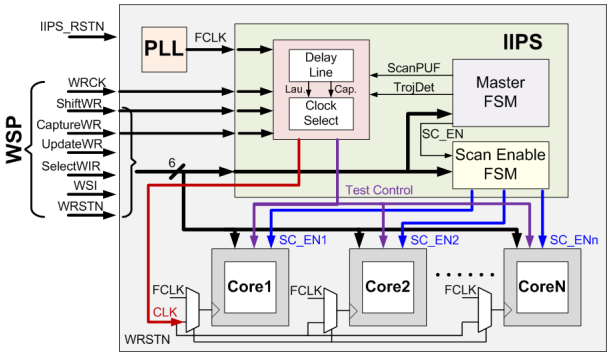

Fig. 3. Block diagram of the IIPS module showing interconnection with other IP cores in a SoC using SoC boundary scan architecture.

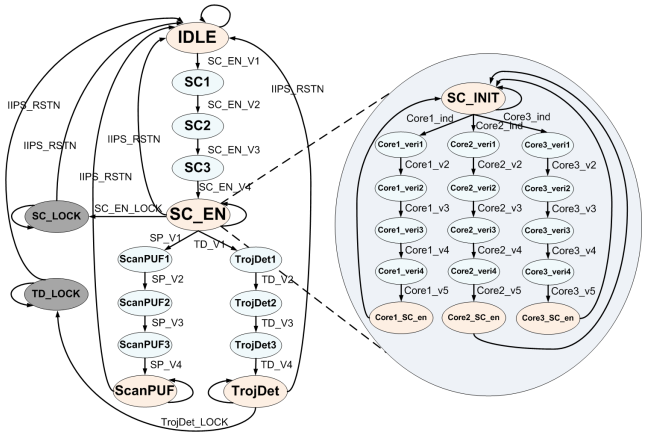

Fig. 4. State transition diagrams of the IIPS master FSM and Scan enable control FSM embedded inside it.

3 Overview of IIPS

The block diagram of IIPS is provided in Fig. 3. It consists of a *Master Finite State Machine* (*M-FSM*) that controls the working mode of IIPS, a *Scan Chain Enabling FSM* (*SE-FSM*) to provide individual control over activation of scan chains in the SoC, and a clock control module to generate necessary clock and control signals for performing *ScanPUF* authentication and path delay based hardware Trojan detection. The state transition diagrams of the *M-FSM* and *SE-FSM* are illustrated in Fig. 4. When enabled, IIPS takes the standard SoC test inputs from Wrapper Serial Port (*WSP*) (*WSO* output pin is not shown). As outputs, IIPS sends one scan chain enable signal to each functional-IP core, and replaces the original test clock *WRCK* with *CLK*, which is generated inside IIPS, to support *ScanPUF* authentication and Trojan detection. Besides, test control *ShiftWR* and *CaptureWR* applied to the cores are replaced with *ShiftWR_TD* and *CaptureWR_TD* during Trojan detection for automatic capture of test responses. In short, IIPS serves as a test control line translator. SoC test inputs arrive at IIPS, and IIPS generates the actual test clock and control signals

that are applied to the cores. During normal testing the test signals are automatically preserved by IIPS. A single functional clock is assumed in this work and multiple clock domains can be adapted with slightly altered design of the clock control module.

After IIPS is turned on, *M-FSM* starts from *IDLE* state. A specific vector sequence applied at *WSP* can bring *M-FSM* to *SC_EN* state to allow scan chain activation. After scan chain activation is done, *M-FSM* can be set to *ScanPUF* or *TrojDet* state to perform *ScanPUF* authentication or hardware Trojan detection, respectively. *SC_EN* state is designed to precede *ScanPUF* and *TrojDet* states, because both *ScanPUF* and Trojan detection require at least one active scan chain. Specific sequences of vectors are required for *M-FSM* to switch among different security functions, acting both as an authentication mechanism and to prevent unintentional trigger of a function. When performing each IIPS function, *M-FSM* stays in the corresponding state; when the task is completed, *M-FSM* can be reset to *IDLE* state with IIPS reset signal *IIPS_RSTN*. The *SC_LOCK* and *TD_LOCK* states are two locking states designed to support scan chain based SoC testing and delay fault testing, respectively, as described below in detail. When IIPS is deactivated or in the locking states, intact test signals are propagated to the functional cores; during IIPS security functions, modified (in *ScanPUF* and *TrojDet*) or deasserted (in *SC_EN*) signals will be applied to the functional cores.

IEEE 1500 compliant SoCs have a standard active low test reset signal *WRSTN*. Since the test inputs *WSP* usually reuse chip functional inputs, *WRSTN* is required to separate the test mode from functional mode by interpreting the inputs as test signals, and consequently configuring the test infrastructure as well as propagating test data. Similarly, one extra input pin *IIPS_RSTN* is needed in the SoC to provide an active low reset signal for IIPS. Active *IIPS_RSTN* disables IIPS by setting *M-FSM* to *IDLE*. *IIPS_RSTN* and *WRSTN* together define four operating modes of the SoC:

(i) {*IIPS_RSTN*, *WRSTN*} = "00", functional mode;
(ii) {*IIPS_RSTN*, *WRSTN*} = "01", basic SoC test mode;
(iii) {*IIPS_RSTN*, *WRSTN*} = "11", advanced SoC test mode requiring faster-than at-speed delay test;
(iv) {*IIPS_RSTN*, *WRSTN*} = "10", IIPS security mode.

In mode (i) and (ii), IIPS is not turned on. Hence mode (ii) can only perform basic SoC testing where core scan chains are disabled, and delay fault testing is not supported. Mode (iii) enables IIPS to allow scan-chain based SoC testing and delay fault testing. One requirement in this procedure is that IIPS should not have unnecessary state transitions in order to avoid interfering with the normal testing procedure. For example, in scan-based testing, IIPS should enable the scan chains as requested and then stay idle during the rest of the test. If *M-FSM* happens to enter *ScanPUF* or *TrojDet* state, the test clock *WRCK* and scan shift signal *ShiftWR* will be tampered, causing wrong test responses. Similarly, during delay testing, IIPS should transition to the *TrojDet* state to provide appropriate clock signals. To achieve such "functional locking", states *SC_LOCK* and *TD_LOCK* are added to *M-FSM* for scan-based and delay testing, respectively. In the locking states, proper output signals are retained, but any state transition is disallowed except for resetting the entire IIPS to *IDLE* with *IIPS_RSTN*. In the *SC_LOCK* state, IIPS maintains the scan chain enable signals as determined in *SC_EN* state. Similarly, the clock and test control signals in *TrojDet* state are available in *TD_LOCK* state for delay testing.

4 Design of IIPS Security Functions

4.1 Attack Models and Mitigation Strategies

4.1.1 Scan-based Attack in Cryptographic Systems

Scan chain is the most prevalent DfT infrastructure in modern ICs. It helps to greatly improve the controllability and observability of internal nodes, thereby enhancing structural test coverage. However, scan chain turns out to be a two-edged sword when it comes to security hardware like cryptographic processors or ASICs. The enhanced controllability and observability that scan chains offer can help reveal internal secret information (e.g. a cryptographic key) from a chip. Previous studies [6] have demonstrated that by operating AES in functional and scan modes alternately, intermediate computation of each clock cycle can be observed through the scan chain. With a chosen plain-text attack, the secret key can then be discovered easily.

Several secure scan architectures have been proposed to prevent access to scan chain by illegal users. The common philosophy behind them is to implement an authentication mechanism to allow only legitimate scan access. In particular, the scan structure in [5] masks the scan output with pseudo random numbers when users fail to embed the correct test key into their test vectors. In [6], authors proposed a method to isolate important data registers (e.g. those holding the keys) when the chip runs in insecure mode. A low-overhead authentication based scan protection was proposed in [7] that gates the scan output unless the authentication is successful.

In IIPS, we adopt *VIm-Scan* proposed in [7] primarily due to the fact that it does not require structural changes on the scan chains except for the minor interface adaption. Hence, it minimizes the design modification in SoC functional cores and facilitates the integration process. It also provides the advantages of ultra-low overhead and good scalability to larger SoCs.

4.1.2 Counterfeiting Attacks

Current trend in outsourcing design and fabrication of SoCs provides opportunities for counterfeiting attacks through cloning and overproduction of chips. Increasing incidences of counterfeit chips in a supply chain pose a serious concern to the SoC designers as well as system integrators. Device authentication has been considered as an effective method to prevent counterfeiting attacks.

By assigning each legitimately fabricated IC a unique ID and registering it in a vendor-built database, consumers can authenticate each IC and thus get only the authentic ones activated and deployed [9]. However, a digitally stored device ID is not secure or tamper-proof. Invasive or non-invasive methods can be applied to reveal the stored ID [10], and a cloned device can be programmed with the same ID subsequently. On the other hand, unclonable IC fingerprints based on PUF have been widely investigated as a reliable vehicle for authentication. PUF forms an important security primitive by exploiting intrinsic process-induced variations in circuit parameters such as path delay to generate unique device signatures. Well-designed PUFs are also expected to exhibit high uniqueness and robustness, thus are capable of reliably identifying authenticity of ICs.

To integrate a PUF primitive into IIPS, it is necessary that it incurs low silicon overhead considering both the PUF circuitry and the control logic for signature generation. Moreover, it is desirable that the signature generation and extraction process complies with standard SoC testing flow. With these requirements, we employ the *ScanPUF* architecture described in [30], which is realized with scan chain in the IPs. It incurs negligible hardware overhead and provides high uniqueness and robustness.

4.1.3 Hardware Trojan Attack

Malicious modification of a design, also known as, hardware Trojan attack, in untrusted fabrication facilities or design house has emerged as a major security concern [34]. Due to their stealthy nature and practically infinite Trojan space (in terms of trigger conditions, forms and sizes), a cleverly inserted Trojan in large SoC is highly likely to evade conventional post-silicon testing. Furthermore, malicious hardware can easily bypass traditional software-implemented defense techniques as it is a layer below the entire software stack.

To address the threat of Trojan attack, several promising Trojan detection approaches have been proposed, including logic testing and side-channel analysis based methods. In the context of IIPS, we focus on the latter methods due to the following reasons. First, it is generally believed that larger Trojans specifically sequential Trojans requiring a sequence of rare events to trigger, are extremely difficult to detect in logic testing, due to difficulty in activating such Trojans [37]. Second, on-chip generation of targeted test vectors is very expensive in terms of hardware resources. Among the various side-channel analysis based detection approaches, combinational path delay based detection [14] has several advantages. First, it does not require triggering the Trojan payload to detect the Trojan. In fact, it does not necessarily require switching activities in the Trojan circuitry, unlike current based Trojan detection approaches. Instead, extra delay due to capacitive loading due to Trojan circuitry can make a Trojan detected. Second, high-resolution path delay measurement can be accomplished through on-chip infrastructure at modest cost, which, can also be used to perform delay fault testing. In this work, we choose the clock sweeping technique proposed in [31] for Trojan detection through path delay measurement using an efficient clock sweeping approach. IIPS provides a centralized infrastructure for characterization of path delays in individual IPs.

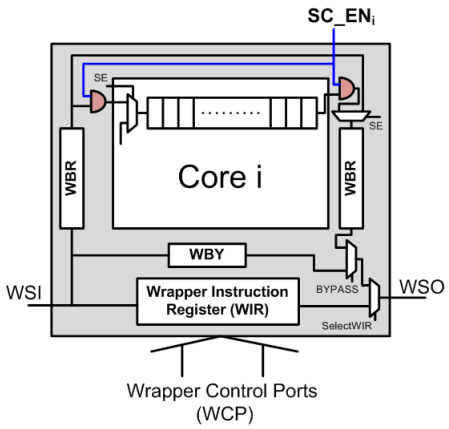

Fig. 5. Functional core scan chain protection with *SC_EN*.

4.2 Design of Security Primitives

4.2.1 Authentication-based Scan Chain Activation

The state diagram of *SE-FSM* is illustrated in the circle in Fig. 4. IIPS generates an external scan chain enable signal SC_EN_i ($i \in [1, N]$, where N is the number of functional cores) for each functional core. The scan chain locking is achieved by gating both the scan input and output when SC_EN_i is inactive. Fig. 5 demonstrates the scan-in and scan-out gating of a IEEE 1500 compliant core. Gating both the input and output of a scan chain can prevent illegal users from gaining controllability and observability via the scan chain. [7] chose to gate only the scan output, which still leaves illegal users the opportunity of scanning in an arbitrary state to operate the chip from, and observe the functional primary outputs. An alternative way of controlling access to the scan chain is to gate the core internal scan enable *SE*. However, this may compromise the timing of *SE*, which is usually one of the critical paths of a SoC.

By default, SC_EN_i is set to logic 0, hence all scan chains are disabled and the cores can only work in functional mode. Even if the scan chain is concatenated with the *wrapper boundary register* (*WBR*) during testing, only logic 0's will be scanned in and shifted out from the scan chain. In order to perform scan-based testing or IIPS security tasks, a test initiation phase is required to enable necessary core scan chains prior to the testing procedure. This can only be done when *M-FSM* is in *SC_EN* state. To enable scan chain access of each functional core, a sequence of input vectors specific to the core has to be provided at test interface *WSP* to bring *SE-FSM* to the particular scan activation state, e.g. *Core1_SC_EN*

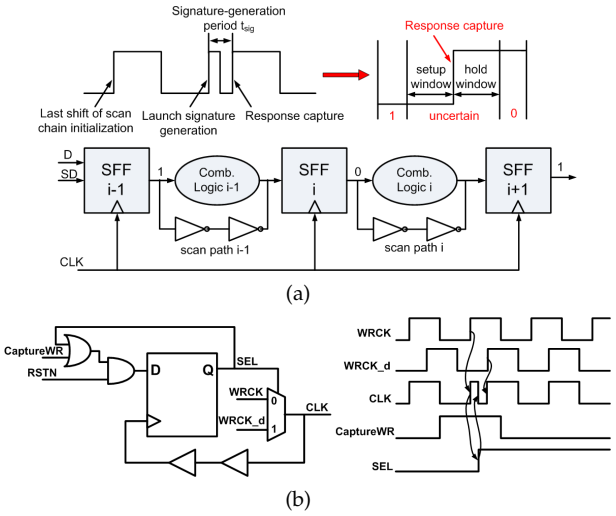

Fig. 6. (a) Signature generation in ScanPUF realized in the scan chains of IPs by the IIPS module; (b) clock generator and timing of relevant signals.

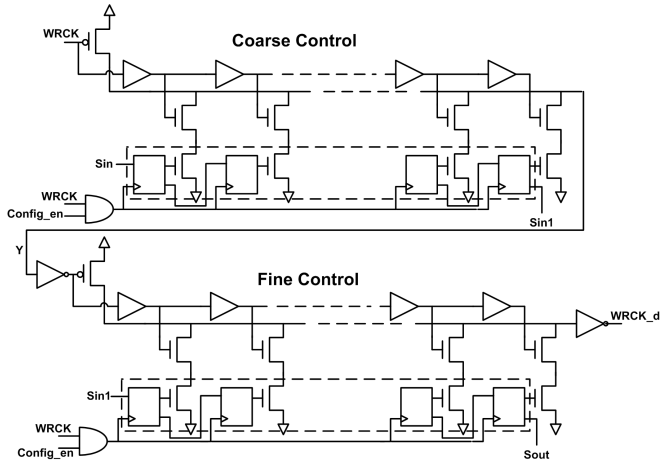

Fig. 7. Design of the programmable delay line [27].

for enabling scan chain of Core1. During scan chain activation, the predefined authentication input sequences are distinct for different scan chains, hence can be considered as the scan *activation key* to each core. After each activation, *M-FSM* will return to *SC_INIT* waiting for the next scan chain activation request. SC_EN_i of the enabled scan chain is latched and can only be disabled by active *IIPS_RSTN*.

4.2.2 ScanPUF Primitive

Fig. 6(a) illustrates the *ScanPUF* concept. The basic idea is to exploit random delay variations in scan paths, i.e. the paths between adjacent scan flip-flops (SFFs). Scan chain is essentially a shift register. When a vector is being shifted in, to guarantee that each SFF can successfully latch the value from the previous SFF, the new value has to be ready by a setup window (t_{setup}) ahead of the rising edge of the clock. This means the shift clock period should be no less than $t_{clk2q} + t_{pd} + t_{setup}$, where t_{clk2q} is the clock-to-Q delay of the SFF and t_{pd} is the combinational propagation delay from the previous SFF, which can include interconnect and buffer delay. Smaller clock period will lead to setup violation, causing uncertain value being latched to the SFF. Generally, scan paths within a functional core can be partitioned into groups, where paths in each group have closely matching delays. In each group, the path delays can be modeled as a single nominal delay (t_{pd0}) plus a random intra-die variation (δ_i, i is the index of the scan path) with Gaussian distribution. By using a shift clock with period $t_{clk2q} + t_{pd0} + t_{setup}$ (denoted as nominal capture clock period t_0), half the SFFs are expected to have a setup failure; and the positions of the failing SFFs in the scan chain depends on the random intra-die delay variations. This creates the PUF signature, i.e. response to a specific capture clock period. In particular, *ScanPUF* signature generation is performed as follows:

(1) *Scan chain initialization*: In test mode (scan-shifting mode), scan-in a sequence of alternating 0's and 1's under normal test clock.

(2) *Signature generation:* Launch a clock rising edge t_{sig} later than the last rising edge in scan chain initialization. Since every SFF is supposed to have a transition, those end up holding the old values indicate setup violations due to longer scan path delay.

(3) *Signature propagation:* Shift out the response bits through scan-out port with normal test clock.

ScanPUF test procedure simply requires scan shift operation with a dedicated clock signal. Therefore, *ShiftWR* should be asserted and applied to the functional cores throughout the signature generation process to enable scan shift. To generate *CLK* with a proper signature capture cycle, test inputs need to acknowledge IIPS on the completion of scan initialization. Since SoC test input *CaptureWR* (one of *Wrapper Serial Control* lines) is not used during SoC configuration or scan shift, it can be used as the acknowledge signal. The actual *CaptureWR* sent to the embedded cores are deasserted because capture operation is not needed during *ScanPUF*. Relevant signal timing in the signature capturing process is demonstrated in Fig. 6(b). The clock generation logic shown in Fig. 6(b) works in the following manner: In the beginning of *ScanPUF* process, *RSTN* is asserted to set *SEL* to 0. Normal test clock *WRCK* is used for scan shifting (*CLK*=*WRCK*), while *ShiftWR* is active during the entire *ScanPUF* process and applied to all the functional cores. *CaptureWR* is asserted before the last shift in scan initialization and holds for one cycle, indicating the completion of scan initialization. In this way, with the clock rising edge of the last shift, *CaptureWR* is latched by a SFF, turning *SEL* to 1 so that *CLK* is switched to *WRCK_d*, which is a phase-shifted version of *WRCK*. The amount of the phase shift is tuned to be the capture

period t_{sig}. Buffers are added before the clock port of the SFF to introduce a slight positive skew, to ensure the capture clock pulse of reasonable width. An overly narrow pulse may cause unsuccessful latching or suffer from distortion when passing through the clock tree. *WRCK_d* is used thereafter until the PUF response is completely shifted out through *WSO*.

As described above, the capture clock is generated by switching between two clocks with a phase difference. The phase shift is realized by adopting a programmable delay line (PDL) proposed in [27], as shown in Fig. 7. It has a coarse delay control and a fine delay control block, differentiated by the delay of a unit buffer. The total delay can be configured by serially loading the control flip-flops to set one bit in each block. The coarse and fine control together allow a fine-grained delay tuning with wide range at low hardware cost. The implementation details are provided in Section 6.1.

For the purpose of authentication, the signature generation is done by the chip manufacturer during production test and by a system integrator before a chip is integrated into a system. The latter generates signature to verify the authenticity and integrity of a chip - i.e. to make sure it is not a counterfeit one. The authentication process is based on off-line testing. The signature does not need to be stored inside a chip for the purpose of authentication. The system designer needs to generate it, extract out through the scan out process and match it with manufacture provided golden signature database. ScanPUF is not designed to target in-field authentication. Therefore, similar to post-manufacturing testing, the ScanPUF authentication by a system designer needs to be done under specified range of test conditions.

4.2.3 Path Delay Based Hardware Trojan Detection

Fig. 8(a) illustrates the basic concept of clock sweeping technique. For a combinational path, it sensitizes it from its source (i.e. a primary input or flip-flop) and captures the output at the destination (i.e. a primary output or flip-flop) with a clock of certain frequency. By sweeping the clock frequency, path propagation delays in a circuit can be categorized into different bins, where each bin stand for a small range of period, determined by the clock sweeping resolution. Efficient test pattern generation to sensitize maximum number of paths can be achieved leveraging the existing CAD algorithms and tools for transient and path delay fault generation [31].

We have implemented a clock generation logic that can generate appropriate clock and test control signals inside the IIPS to support both Launch-On-Capture (LOC) and Launch-On-Shift (LOS) test schemes. In clock sweeping based Trojan detection, Trojan impacts are modeled as path delay variations. The testing responses over the entire circuit are collected for statistical analysis to identify possible Trojans. We make the capture clock frequency tunable to support a fine-grained frequency sweeping over a wide range, thus covering non-critical paths even

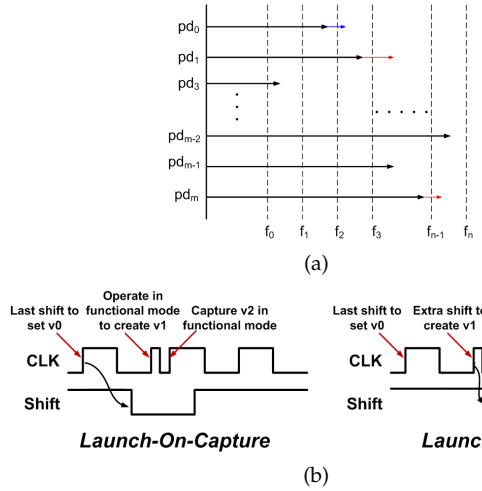

Fig. 8. (a) Concept of clock sweeping technique for hardware Trojan detection; (b) Trojan detection through monitoring of delay shift by observing the latched value under clock sweep in two possible schemes.

with short delays. Path delay characterization is a two-pattern test, which involves one vector to initialize the inputs and a second one to excite the transition on the target path. LOC and LOS refer to two ways of generating the second test vector. LOC takes the functional result of the first vector (V_0) as the second one (V_1), namely

$$V_0 = \{PI_0, PPI_0\} \quad (1)$$
$$V_1 = \{PI_0, f_{PPI}(V_0)\} \quad (2)$$

where PI represents the primary input, PPI stands for the pseudo primary inputs, i.e. internal state elements, and f_{PPI} is the boolean dependency of PPI on $\{PI, PPI\}$. In this case, the circuit should operate in functional mode during both the V_1 generating cycle and the capture cycle, hence the scan enable (*Shift*) can be deasserted before the V_1 generating cycle and reasserted after the capture cycle, as shown in Fig. 8(b). In this way the scan enable timing just needs to meet the test clock frequency, and does not depend on capture cycle. On the contrary, LOS scheme shifts the first test vector by one bit to create the second one, which poses a stringent requirement on the scan enable timing, as shown in Fig. 8(b). The turnaround time of the scan enable is limited by the capture clock period.

Fig. 9 demonstrates the clock generator circuitry as well as the timing of relevant signals. The basic clock generation scheme is similar to that of *ScanPUF*. However, in this case both *ShiftWR* and *CaptureWR* are used to control IIPS, and the actual test control lines *ShiftWR_TD* and *CaptureWR_TD* that go to the cores are generated by IIPS. Particularly, *ShiftWR* is used to acknowledge the completion of test initialization by holding low for one cycle after the last shift for V_0. *CaptureWR* is used to differentiate LOC (*CaptureWR*=1) from LOS (*CaptureWR*=0).

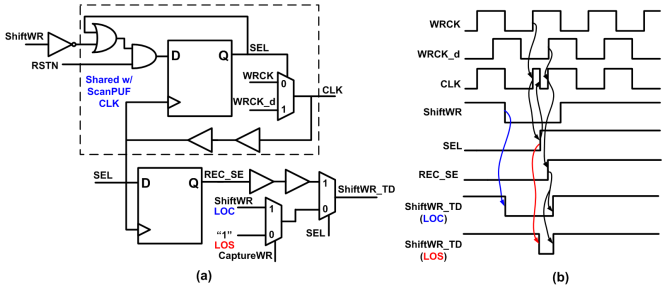

Fig. 9. Clock sweeping based hardware Trojan detection: (a) clock generator; (b) clock generation signal timing.

During the clock generation, rising of *SEL* indicates occurrence of the launching clock edge, and rising of *REC_SE* indicates the capture clock edge. While for both LOC and LOS, the scan enable *ShiftWR_TD* returns to 1 with *REC_SE* after the capture, *ShiftWR_TD* in LOC is deasserted together with *ShiftWR* to allow the core to enter functional mode in order to generate the launching vector, and *ShiftWR_TD* in LOS maintains high until *SEL* rises with the launching clock edge.

Trojan detection covering every region of a SoC is time-consuming and prone to false positive results. Hence, the side-channel analysis should ideally target security-critical regions of a SoC to make it more effective. For a given SoC, designers can figure out critical IPs in a SoC or critical regions of an IP (e.g. memory management unit, crypto cores, buses). Such an approach will not increase the hardware overhead of IIPS. However, it can significantly reduce the Trojan detection time and cost.

5 Test Protocol under IEEE Std. 1500

5.1 Wrapper Operation Modes

IEEE Std. 1500 allows testing of a SoC in various configurations via collaborative control of *Wrapper Instruction Register* (*WIR*) and *Wrapper Serial Control* (*WSC*) lines. It can concatenate the core boundary registers and scan chains of embedded cores into a test path, thus enabling test access to each core. Fig. 10(a) demonstrates the concatenated *WBR*s of SoC cores. By loading the *WIR* with different instructions, the test path can include or exclude the scan chain, and the decision can be core specific. As shown in Fig. 10(a), Core 2 is incorporated into the test path with its internal scan chain, allowing testing of the core in scan mode. IEEE Std. 1500 supports core based testing, namely when a core is under test, irrelevant cores can be set into *BYPASS* mode with only the 1-bit *WBY* register included in the test path so as to save test time. This can be done by configuration of *WIR*s of irrelevant cores with *BYPASS* instruction.

Different core operation modes include functional mode, inward facing mode, outward facing mode, and bypass mode. Functional mode corresponds to the scenario where the SoC is performing its normal function, while the other three modes are for testing. In particular, inward facing mode allows testing the internal logic of a core; outward facing mode supports validating the functionality of interconnects among different cores; and bypass mode indicates use of *WBY* register to facilitate testing of other cores. These operation modes are realized by configurations of the *Wrapper Boundary Cell* (*WBC*), i.e. element in *WBR*.

TABLE 1
Control values for wrapper boundary cell

	Input Cell		Output Cell	
Instruction	*Scan Enable*	*Hold Enable*	*Scan Enable*	*Hold Enable*
WS_INTEST	*ShiftWR*	1	*ShiftWR*	$\sim$ *CaptureWR*
WS_EXTEST	*ShiftWR*	$\sim$ *CaptureWR*	*ShiftWR*	1

Fig. 10(b) displays the structure of a minimal *WBC* and its configurations to realize different operations. The operation mode a *WBC* performs, controlled by values of *Scan Enable* and *Hold Enable*, depends both on the core configuration (i.e. instruction in *WIR*) and the test control lines *WSC*. IEEE 1500 serial test interface contains a test data input terminal *WSI* and output terminal *WSO*, a test clock *WRCK*, a test reset *WRSTN*, and test control lines *WSC*. Mandatory *WSC* signals include *SelectWIR* which, when active, connects *WSI* with *WIR* for test mode configuration; *ShiftWR* controls the shift operation of *WBR* and *CaptureWR* indicates capture operation of the wrapper cells. The actual dependency of *Scan Enable* and *Hold Enable* on *WSC* is mode dependent. Example implementation of *WBC* control is shown in Table 1 for inward facing and outward facing modes [36].

5.2 SoC-Level IIPS Test Protocol

5.2.1 Scan Chain Authentication

The control of IIPS *M-FSM* and *SE-FSM* is at SoC level. Therefore, for overall IIPS control and scan chain activation, creation of test protocols/patterns is the system integrator's responsibility. Test can be directly provided in terms of SoC primary inputs and delivered in the form of standard Core Test language (CTL) [24]. In particular, the enabling process of each security mode (i.e. *SC_EN*, *ScanPUF* or *TrojDet*) can be modeled as a test macro.

5.2.2 ScanPUF

On the other hand, it is desirable to generate test sets for *ScanPUF* and Trojan detection at core level by core designers, because *ScanPUF* test data lengths and Trojan detection test patterns are core dependent. The test sets can then be expanded to SoC level by the system integrator. For test sets in conventional testing, expansion from core to SoC level involves translating the core terminals to the corresponding SoC pins, as well as searching for paths to and from the CUT for test stimuli and response propagation when using the parallel test interface [28]. Expansion of *ScanPUF* and Trojan detection test sets can reuse the same methodology, with the main distinction

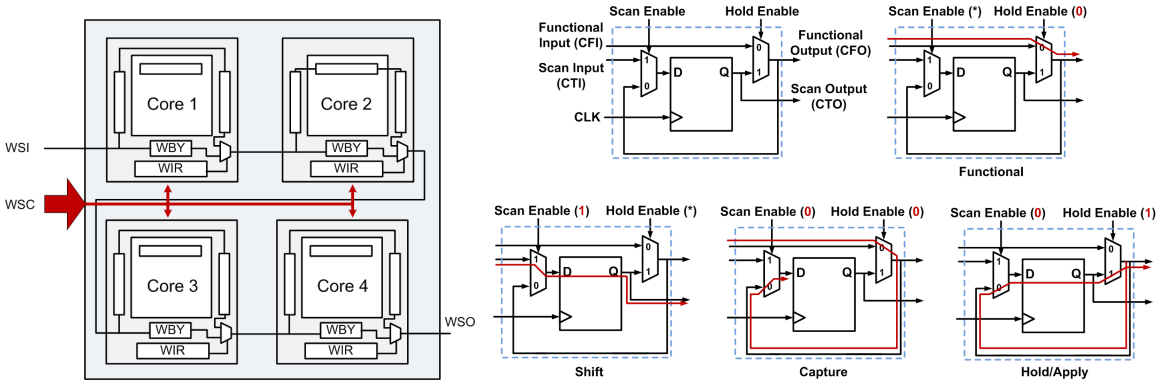

Fig. 10. (a) Wrapper boundary registers concatenated for SoC test or IIPS functions; (b) wrapper boundary cell and its various configurations.

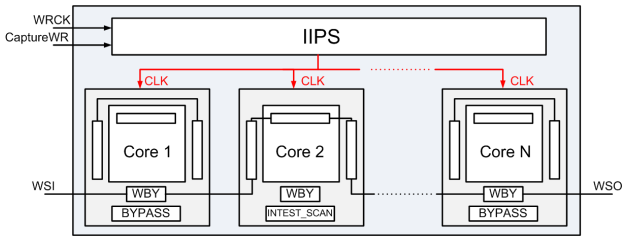

Fig. 11. Configuration of scan chains of different IPs during signature generation for chip authentication using scanPUF structures.

that an acknowledge signal should be raised for one clock cycle at the completion of test initialization.

For both *ScanPUF* and Trojan detection, the entire test procedure can be divided into four parts: (1) core test mode configuration; (2) test initialization; (3) signature generation; and (4) signature propagation. In case of *ScanPUF*, although scan chains in SoC functional cores can be concatenated into a single scan chain to generate *ScanPUF* signature, scan paths in different cores may have different mean delays and require different t_{sig}. Hence, an efficient way for saving test time is to perform *ScanPUF* test on different cores individually by setting the Core Under Test (CUT) to *INTEST_SCAN* mode (i.e. connect both *WBR* and core internal scan chain into the test path) while configuring other cores to *BYPASS* mode, as shown in Fig. 11. This procedure is referred to as core test mode configuration, which conforms to the procedure followed in normal SoC testing. We note that core-based signature generation can effectively obtain information from all scan chains but in a per-core fashion. Even for a short scan chain (e.g. 256 bit), we can generate a signature of adequate length (e.g. 128 bit) for authentication. However, a longer scan chain gives us the opportunity to choose the scan paths in many possible ways, thus providing more challenge-response pairs, which can increase security for authentication or can be used to generate keys for crypto operations.

Within an IP, a long scan chain is usually broken into a hierarchy of multiple shorter scan chains to improve the test time. Such a scan architecture, however, would not affect the uniqueness of the proposed scanPUF implementation. Actually, multiple scan chain would help in accelerating the pre-load of scan FFs and scan out of the signature.

ScanPUF test initialization refers to initializing the entire scan chain with alternate 0's and 1's. Beside test data input and output translation, the initialization data length is expanded to account for the *BYPASS* registers of irrelevant cores on the test path. On completion of test initialization, SoC input *CaptureWR* is asserted for one clock cycle to launch the signature generation and propagation process. Thereafter, one maintains the test lines and pausing the clock until the signature is shifted out. We note that the actual clock and *WSC* signals applied to the cores for test control will be automatically generated by IIPS, the only responsibility at the SoC boundary is to raise the acknowledge signal when test initialization ends.

5.2.3 Hardware Trojan Detection

For Trojan detection, test set creation can be considered in two scenarios. To detect Trojans mounted inside the functional cores, test sets should be developed by the core designers and then expanded to SoC level. Trojans could also be inserted to tamper the interconnection among cores, e.g. on system buses. To test against such attacks, test sets should be created at SoC level.

Fig. 12 illustrates the core configurations in the two scenarios. In the first case, to detect Trojans inside a core, the CUT is set to *INTEST_SCAN* mode to allow inward facing test, while other cores are set to *BYPASS* mode. Here test initialization means scanning in test vectors for CUT primary and pseudo primary inputs (scan flip-flops) before launching the capture clock. Associated with this step is the translation of the test terminals

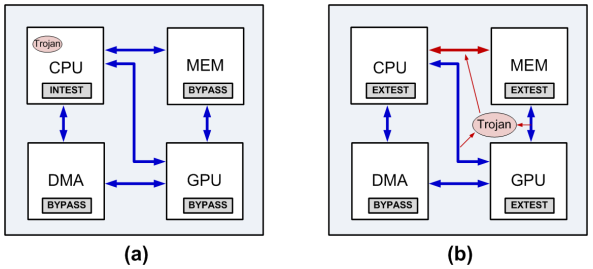

Fig. 12. Example SoC configuration when performing Trojan detection for: (a) Trojans inside a core; (b) Trojans in SoC system bus.

from core to SoC boundary, and expansion of the test data to account for the *BYPASS* registers in other cores, which is trivial because test is executed through the serial interface. In the second case where integrity of system interconnections is concerned, test patterns can be directly created at SoC level and no test data expansion is needed. Multiple test sets can be generated, with each focusing on Trojan detection of a particular region of interconnects. When one region of interconnect is being tested, the corresponding driver and receiver cores should be configured to *EXTEST* mode while other cores in the *BYPASS* mode. In this way, output wrapper registers of the driver cores can provide test stimuli, and test responses can be captured at receiver core input wrapper registers.

The test scheme choices are also different for the two cases. When considering Trojans inside a core, both LOC and LOS can be used to capture test response. However, for Trojans on system buses, only LOS scheme can be applied for minimally configured IEEE 1500 architecture. This is because minimally configured output wrappers do not support capture operation in *EXTEST* mode, thus cannot generate the excitation pattern in functional mode with LOC. However, with higher 1500 configuration where output wrappers are implemented with enhanced scan cells, or extended 1500 infrastructure to support output wrapper capture in *EXTEST* mode [26], LOC scheme can also be used for detecting Trojans on system interconnects. A simplified example of SoC level test protocol for Trojan detection using LOC is represented in Algorithm 1. It is worth noting that *WSC* may have patterns that are invalid for normal testing, as the actual test control lines are generated by IIPS.

Algorithm 1 SoC-level test protocol for Trojan detection

```
MacroDefs Core_TrojDet_macros{
    //core test mode configuration
    core_config{
    // If core is CUT, configure to WS_INTEST_SCAN
    // Otherwise WS_BYPASS (Omitted here)
      Purpose Instruction;
      W core_config_timing;
      C {WRSTN=0; WSI=0; WSO=x; WSC=0000;}
      V {WRSTN=1; SelectWIR=1;} //to load WIR
      Shift {
      V {WSI=010;} //instruction WS_INTEST_SCAN
      }
      V {WSC=0001;} //update WIR
    }
    //Trojan Detection
    TrojDet{
    //Test initialization
      W TrojDet_timing;
      C {WRSTN=1; WSC=0110; } //enable scan, LOC
      Shift {
        V {WSI='wsi'; WSO=#; } //apply test pattern
      }
    //Signature generation acknowledge
      V {ShiftWR=0; }
    //Signature propagation
      F {ShiftWR=1; }
      Shift {
        V {WSI=0; WSO='wso';} //check response
      }
    }
}
```

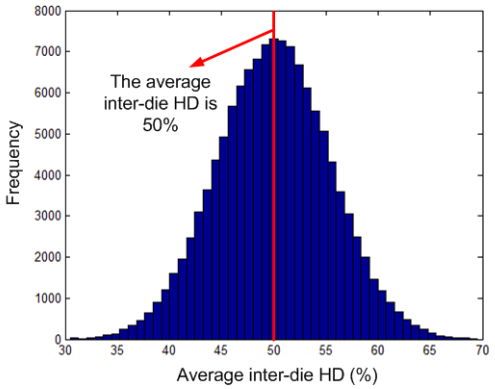

Fig. 13. Inter-die HD distribution for scan-based PUF in 500 chips under $t_{sig} = 0.19ns$

6 Results

To verify the functionality and security effectiveness of IIPS, we performed HSPICE simulation on a IIPS module using 45 nm CMOS process model. We also implemented IIPS in a representative SoC on Altera DE0 FPGA platform. Functional simulations validated the IIPS state transitions in response to SoC test inputs and the timing of the clock generator module. We also analyzed the hardware overhead in both ASIC and FPGA contexts.

6.1 SoC Authentication and Trojan Detection

The clock generator including the PDL block is implemented using 45 nm CMOS Predictive Technology Model (PTM) [29]. The PDL contains a 6-level coarse delay control and a 11-level fine delay control block, with reso-

lution of $136ps$ and $12ps$, respectively. The tunable clock has period range of $\sim 300ps - 1.15ns$, i.e. $\sim 870MHz - 3.33GHz$ frequency. Monte Carlo HSPICE simulations were performed to verify the effectiveness of IIPS in supporting ScanPUF and delay based Trojan detection. To evaluate uniqueness of signatures generated by the PUF, we employed the inter-die Hamming distance (HD) as a measure of the difference between two signatures. Fig. 13 shows the inter-die HD histogram of signatures (128 bits each) in 500 chips when the signature-generation period is 0.19 ns. We can observe that most of HDs are around 50%, which means that nearly 64 bits in a signature are different from others. Hence, the ScanPUF in IIPS can provide excellent performance in SoC authentication with a highly unique signature in each SoC. [30] also shows that under temporal variations (temperature or supply voltage fluctuation), signature of ScanPUF has good robustness with only a few flipped bits.

For Trojan detection, Monte Carlo HSPICE simulations were performed on combinational paths starting from and ending at flip-flops. A test path example is provided in Fig. 14. Since the biggest challenge in side-channel analysis based Trojan detection is process variations that can significantly mask the Trojan effect, it is necessary to validate IIPS Trojan detection capability under inter-die and intra-die process variations. The smallest Trojan that can be reliably detected in terms of its impact on path delay is used to indicate IIPS Trojan sensitivity. In particular, two combinational paths of delay $393ps$ and $1013ps$ were chosen, representing short and long paths in real circuits, respectively. Inter-die variations of transistor threshold voltage V_{th} with standard deviation $\sigma_{th,inter} = 5\%, 10\%$, and 15% were considered, with intra-die variation of standard deviation $\sigma_{th,intra} = 5\%$. At each process corner, a test is first performed on one Trojan-free path to find out the capture frequency, which is then used to differentiate Trojan-infected paths from Trojan-free ones. We inserted multiple Trojans incurring different delays at each process corner to determine the Trojan sensitivity. We consider reliable detection of Trojan as a miss rate of less than 5%, including false positive and false negative. For each process corner and each path, 100 Trojan-free and 100 Trojan-infected copies of the path with V_{th} following Gaussian distribution were simulated. Fig. 15 demonstrates IIPS Trojan sensitivity at each process corner. It can be observed that the Trojan detection resolution decreases slightly with increasing inter-die process variations, and is lower for the longer path. This is because the over all impact of variation on the long path is greater than that on the short path, mainly due to higher logic levels. With $\sigma = 5\%$ intra-die and $\sigma \leq 15\%$ inter-die process variations, Trojans causing delay above $44ps$ can be reliably detected, equivalent to the delay caused by an extra level of XOR gate.

It is worth noting that Trojan detection approaches generally rely on statistical methods, e.g. principal component analysis [14] or multidimensional scaling [31], to isolate the Trojan-infected ICs from process noise with high confidence. In this paper, our focus is to show that IIPS module can facilitate Trojan detection at IP and SoC level using on-chip infrastructure. The proposed path delay analysis based Trojan detection can, however, be combined with statistical analysis to significantly improve Trojan detection sensitivity and reliability.

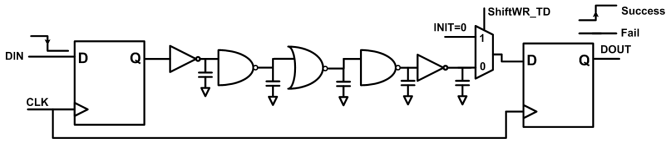

Fig. 14. Example combinational path used as a model of Trojan attack.

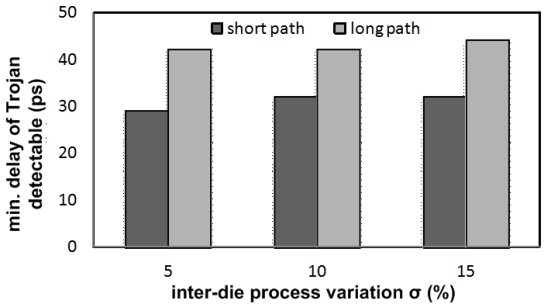

Fig. 15. Minimum delay of Trojan detectable with clock sweeping technique implemented in IIPS for $\sigma = 5, 10, 15\%$ intra-die process variation.

6.2 Hardware Overhead

We use two representative SoC benchmarks as the baseline design to characterize the silicon overhead introduced by IIPS. SoC benchmark 1 contains three ISCAS89 sequential circuits, while SoC benchmark 2 tries to imitate realistic designs by incorporating a 32-bit DLX processor, an AES with 128-bit key, and a 128-bit pipelined FFT module. Both designs have full-scan infrastructure. The numbers of scan chains are estimated by assuming an average scan chain length of 250, which is obtained from the first ten benchmark circuits in ITC02 SoC benchmark [33]. We note that although the *SE-FSM* scales up with increasing number of scan chains, the number of state elements in *SE-FSM* increases with only the rate of $log_2(C \cdot p)$, i.e. $log_2(p) + log_2(C)$, where p is the rate of increasing number of scan chains and C is the authentication key length. Table 2 demonstrates that percentage area overhead induced by IIPS decreases remarkably with the size of the benchmark circuits. For real designs, the overhead is less than 1%. In addition, since IIPS resides outside the functional modules and is used only when performing off-line testing or security tasks, it does not incur functional performance degradation or power overhead. The clock tree routing can also be preserved by keeping multiplexers between IIPS clock and functional clock inside IIPS.

TABLE 2
Hardware overhead of IIPS w.r.t. two example SoCs

Cores/SoC	Benchmark SoC 1					Benchmark SoC 2				
	IP Cores			SoC	IIPS	IP Cores			SoC	IIPS
	s1423	s5378	s9234			DLX	AES	FFT		
Area (μm^2)	1151	2659	3344	7154	546 ***(+7.6%)***	30845	83049	500386	614280	4978 ***(+0.8%)***
# of SFFs	74	179	211	464	-	521	2469	33203	36193	-
# of scan chains	1	1	1	3	-	3	10	133	146	-

TABLE 4
Hardware Trojan detection results on FPGA

Path index	1	2	3	4	5	6	7	8	Average
Path delay (ns)	3.7	4.5	5.8	6.1	7.2	8.1	10.3	13.1	-
Trojan induced delay overhead (ns)	0.37	0.26	0.46	0.30	0.26	0.33	0.59	0.19	0.35
Miss rate	0	9.4%	3.1%	9.4%	3.1%	12.5%	0	12.5%	6.3%

6.3 Experimental Validation

Hardware validation of IIPS was performed on a FPGA platform where Altera DE0 FPGA development boards were used to emulate the ASIC scenario. IIPS was implemented on 65 nm Cyclone III FPGA devices used in the DE0 boards. Due to lack of precise control over the placement and routing of the mapped design, the PDL cannot be implemented as in ASIC to produce precisely tunable delays. We employ the on-chip Phase Locked Loop (PLL) module in the FPGA to generate the phase shifted clock. The on-chip PLL can provide phase shift at resolution of 97ps in our experiments, which is equivalent to the clock frequency sweeping resolution. We have implemented a minimally configured IEEE 1500 infrastructure in SoC benchmark 1, and validated the functionality of IIPS by interfacing it with the benchmark circuit. The hardware overhead incurred by IIPS is provided in Table 3. Part of the additional resource utilization is contributed by the control logic for PLL, which should not be required in ASIC scenarios. To test the effectiveness of the clock generation logic, we performed Trojan detection with IIPS implemented in SoC benchmark 1. In particular, Trojans were inserted on 8 different paths of s9234 core. A Trojan was realized as a single inverter inserted at different locations of the paths to model minimal delay impact of an extra logic level. The underlying assumption is that the payload of a typical digital Trojan will introduce at least one extra logic gate on certain path of the circuit. We preserved the post-fitting netlist of the Trojan-free design by using incremental compilation to insert the Trojan to assure the extra delay is caused by Trojan logic rather than changes in global layout topology. Target circuit paths were chosen to have nominal delay ranging from 3.7 ns to 12.5 ns. 16 FPGA boards were used, with Trojan-free and Trojan-infected design mapped once on each board for 32 copies of the design. The percentage of incorrect detection (*miss rate*) for each path is provided in Table 4. It can be observed that the minimal single-inverter Trojan with average delay of 0.35ns can be reliably detected with miss rate of 6.3%. The miss rate is expected to increase monotonically with increasing path length. Note that, the measured miss rate varies due to minor differences in Trojan placement and routing that are out of our control. The actual Trojan coverage can be much higher using statistical analysis, as explained in Section 6.1.

TABLE 3
IIPS overhead in FPGA platform

	SoC 1	SoC 1 w/ IIPS	Overhead
# of LUTs	1194	1238	+3.7%
# of Registers	533	577	+8.3%

7 Discussion

7.1 Flexibility

The IIPS test protocol described in this paper is based on a minimal implementation of IEEE 1500 test infrastructure. However, IIPS functions are flexible in interfacing with enhanced configurations of IEEE 1500 architecture. In fact, IIPS test efficiency can be improved with advanced features of IEEE 1500. Both *ScanPUF* and Trojan detection test protocols can be adapted to use the parallel interface for test vector propagation to save test time, when a parallel TAM architecture is available. For the minimally configured IEEE 1500 infrastructure, Trojan detection on system buses can only be performed with LOS scheme, because core output wrapper cells do not support a capture operation in *EXTEST* mode. However, if the wrapper is implemented using *WBC*s with enhanced scan functions, or the infrastructure is extended to support delay-fault test [26], a broadside capture can be realized at the system bus inputs for LOC test scheme. It would improve both hardware Trojan as well as delay-fault coverage.

Furthermore, in the case where a SoC is equipped with a different test wrapper interface than IEEE 1500, the integration and/or design of IIPS can be adjusted accordingly. Since most test wrappers would provide

access to the scan and internal test infrastructure of an IP, as required by IIPS, it can be easily extended to another test wrapper interface.

7.2 Scalability

IIPS exhibits good scalability when applied to large complex SoCs. Since the ScanPUF based authentication and Trojan detection procedures are core-based, the test time either does not increase (for authentication), or increase linearly with the SoC size (for Trojan detection). Protection against scan based attack is based on system-level authentication and scan chain activation logic is independent of the SoC size. The hardware overhead is minimal since IIPS utilizes the existing IEEE 1500 DFT infrastructure. This justifies our observation in Table 2 that the percentage overhead is very small for larger SoCs.

IIPS also exhibits good functional scalability in terms of integrating more security functions or test infrastructures at minimal extra overhead. It can be adapted to provide protection against other attack models, e.g. side-channel attack on crypto cores. For example, IIPS can be equipped with a noise injector [32] to mask key-related information leak through transient current. Moreover, IIPS can incorporate multiple countermeasures for certain attacks to enhance the overall security of a SoC. For instance, a current monitor can be integrated to detect behavior of hardware Trojans at run-time [34]. IIPS can also include an assertion-based security checker to detect malicious inclusions in on-chip processors [35]. Moreover, it can potentially be integrated with other infrastructure IPs - e.g. test infrastructure IPs which provide built-in self-test (BIST) capability to SoCs [19]. Integrating security functions and test support logic in IIPS can significantly minimize the design and hardware overhead due to resource sharing and a centralized control logic. IIPS can support a library of security functions that provides a system integrator design-time configurability to selectively enable/disable a function based on design requirements.

7.3 Integrity of IIPS

Security of the IIPS itself against tampering or unauthorized access is important to ensure its effectiveness. While the IPs that go into a SoC design can be acquired from 3rd party untrusted vendors, the IIPS module itself can be designed by the SoC manufacturer in a trusted environment. However, malicious modification of IIPS can possibly be made in an untrusted foundry. The IIPS design would primarily include FSMs, which would be extremely difficult to reverse engineer from their layout. Extraction of state machine of reasonable size from a synthesized FSM is known to be an intractable problem. Hence, incorporating hard-to-detect malicious modification in these FSMs can be practically infeasible.

Furthermore, possible Trojan attacks in the sequential logic in IIPS (both in foundry and design house) can be reliably detected through efficient side-channel analysis, such as temporal self-referencing, which aim at detecting any modification to a well-defined state transition function [37]. Finally, to enhance the security of IIPS against deliberate malicious modification, one can employ low-cost hardware obfuscation approaches [12], that makes reverse engineering and Trojan attacks significantly more difficult to mount.

8 Conclusion

We have presented a novel paradigm of secure SoC design using an infrastructure IP, referred to as IIPS. IIPS is a centralized low-overhead on-chip module that interfaces with other IPs in SoC and provides countermeasures against various security threats. It features ease of integration, compliance to standard SoC test protocol, functional flexibility and scalability. We have shown the effectiveness of an IIPS module that incorporates a low-overhead authentication mechanism for preventing scan-based attacks; a PUF to protect against piracy and counterfeit chips; and a test infrastructure for trust validation against hardware Trojan attacks. We have validated the functionality and security of IIPS through circuit-level simulations as well as experimental measurements on an FPGA platform. Both ASIC and FPGA implementations of IIPS show that it incurs ultra-low hardware overhead. IIPS can flexibly interface with SoCs equipped with various configurations of IEEE 1500 infrastructure, and can benefit from higher configurations in test efficiency. IIPS can be effectively integrated with test infrastructures to minimize the design cost and overhead. It can be extended to implement protection against other attacks e.g. side-channel attacks at the SoC level. Future work will include effective integration of IIPS with other infrastructure IPs for test and debug as well as expansion of its capability to protect against other attacks.

References

[1] S. Mangard, E. Oswald and T. Popp, "Power Analysis Attacks-Revealing the Secrets of Smart Cards," Springer 2007.

[2] K. Tiri and I. Verbauwhede, "A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation," *Design, Automation and Test in Europe*, 2004.

[3] Y. Lu, M.P. O'Neill and J.V. McCanny, "FPGA Implementation and Analysis of Random Delay Insertion Countermeasure against DPA," *Intl. Conference on ICECE Tech.*, 2008.

[4] G.E. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," *Design Automation Conference*, 2007.

[5] J. Lee, M. Tehranipoor, C. Patel and J. Plusquellic, "Securing Scan Design Using Lock & Key Technique," *IEEE Intl. Symp. on Defect and Fault Tolerance in VLSI Systems*, 2005.

[6] B. Yang, K. Wu and R. Karri, "Secure Scan: A Design-for-Test Architecture for Crypto Chips," *Design Automation Conference*, 2005.

[7] S. Paul, R.S. Chakraborty and S. Bhunia, "VIm-Scan: A Low Overhead Scan Design Approach for Protection of Secret Key in Scan-Based Secure Chips," *IEEE VLSI Test Symp.*, 2007.

[8] F. Koushanfar, "Integrated Circuits Metering for Piracy Protection and Digital Rights Management: an Overview," *ACM GLS-VLSI*, 2011.

[9] V.V.D. Leest and P. Tuyls, "Anti-counterfeiting with hardware intrinsic security," *Design, Automation and Test in Europe*, 2013.

[10] B. Gassend, D. Clarke, M.V. Dijk and S. Devadas, "Delay-Based Circuit Authentication and Applications," *ACM Symposium on Applied Computing*, 2003.

[11] D. Hely, M. Flottes, F. Bancel, B. Rouzeyre, N. Berard and M. Renovell, "Scan Design and Secure Chip," *IEEE Intl. On-Line Testing Symp.*, 2004.

[12] R.S. Chakraborty and S. Bhunia, "Security Against Hardware Trojan Attacks Using Key-based Design Obfuscation," *Journal of Electronic Testing: Theory and Applications*, vol. 27. no. 6, pp. 767-785, Dec 2011.

[13] X. Zhang and M. Tehranipoor, "RON: An on-chip ring oscillator network for hardware Trojan detection," *Design, Automation and Test in Europe*, 2011.

[14] Y. Jin and Y. Makris, "Hardware Trojan detection using path delay fingerprint," *IEEE Symp. on Hardware Oriented Trust and Security*, 2008.

[15] H. Salmani, M. Tehranipoor and J. Plusquellic, "A Novel Technique for Improving Hardware Trojan Detection and Reducing Trojan Activation Time," *IEEE Transactions on Very Large Scale Integration Systems*, Vol. 20, No. 1, Jan 2012.

[16] Synopsys Verification IP: http://www.synopsys.com/Tools/Veri-fication/FunctionalVerification/VerificationIP/Pages/default.aspx

[17] Intellitech Test-IP Product Family: http://www.intellitech.com/products/boundary_scan_test.asp

[18] Y. Zorian, "Guest Editor's Introduction: What is Infrastructure IP?" *IEEE Design & Test of Computers*, 2002.

[19] P. Bernardi, M. Rebaudengo and M.S. Reorda, "Exploiting an I-IP for in-field SoC test," *IEEE Intl. Symp. on Defect and Fault Tolerance in VLSI Systems*, 2004.

[20] S. Tabatabaei and A. Ivanov, "Embedded Timing Analysis: A SoC Infrastructure," *IEEE Design & Test of Computers*, 2002.

[21] E. Dupont, M. Nicolaidis and P. Rohr, "Embedded Robustness IPs for Transient-Error-Free ICs," *IEEE Design & Test of Computers*, 2002.

[22] J. Bordelon, B. Tranchina, V. Madangarli and M. Craig, "A Strategy for Mixed-Signal Yield Improvement," *IEEE Design & Test of Computers*, 2002.

[23] F. DaSilva *et al.*, "Overview of the IEEE P1500 Standard," *Intl. Test Conference*, 2003.

[24] IEEE 1450.6 Core Test Language (CTL): http://grouper.ieee.org/groups/ctl/

[25] IEEE 1500 Embedded Core Test: http://grouper.ieee.org/groups/1500/

[26] Q. Xu and N. Nicolici, "Delay Fault Testing of Core-Based Systems-on-a-Chip" *Design, Automation and Test in Europe*, 2003.

[27] R. Tayade and J.A. Abraham, "On-chip Programmable Capture for Accurate Path Delay Test and Characterization," *Intl. Test Conference*, 2008.

[28] E.J. Marinissen, "The Role of Test Protocols in Automated Test Generation for Embedded-Core-Based System ICs," *Journal of Electronic Testing: Theory and Applications*, Vol. 18 Issue 4-5, Aug.-Oct., 2002.

[29] Available Online: http://ptm.asu.edu/.

[30] Y. Zheng, A.R. Krishna and S. Bhunia, "ScanPUF: Robust Ultralow Overhead PUF Using Scan Chain," *Asia and South Pacific Design Automation Conference*, 2013.

[31] K. Xiao, X. Zhang and M. Tehranipoor, "A Clock Sweeping Technique for Detecting Hardware Trojans Impacting Circuits Delay," *IEEE Design & Test of Computers*, Issue 99, 2013.

[32] X. Wang, W. Yueh, D.B. Roy, S. Narasimhan, Y. Zheng, S. Mukhopadhyay, D. Mukhopadhyay and S. Bhunia, "Role of Power Grid in Side Channel Attack and Power-Grid-Aware Secure Design," *Design Automation Conference*, 2013.

[33] ITC'02 SoC Test Benchmarks: http://itc02socbenchm.pratt.duke.edu/

[34] 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 of Computers Special Issue on Smart Silicon*, 2012.

[35] M. Bilzor, T. Huffmire, C. Irvine and T. Levin, "Security Checkers: Detecting processor malicious inclusions at runtime," *IEEE Symp. on Hardware Oriented Trust and Security*, 2011.

[36] S. Francisco *et al.*, "The Core Test Wrapper Handbook," Springer 2006.

[37] S. Narasimhan, X. Wang, D. Du, R.S. Chakraborty and S. Bhunia, "TeSR: A Robust Temporal Self-Referencing Approach for Hardware Trojan Detection," *IEEE Symp. on Hardware Oriented Trust and Security*, 2011.

9 Author's Bio

Xinmu Wang received her PhD degree in computer engineering at Case Western Reserve University, Cleveland, OH in 2014. Currently she is a product security engineer at Qualcomm Inc., San Diego, CA. She has research interests in hardware security, including hardware Trojan design and detection, design for cryptographic system side-channel attack resistance, and secure SoC design.

Yu Zheng received the B.S. degree in communication engineering and M.S. degree in communication and information system from University of Electronic Science and Technology of China (UESTC) in 2007 and 2010, respectively. He is currently pursuing the Ph.D. degree in computer engineering at Case Western Reserve University, Cleveland, OH. He held an internship in LSI Corporation during the summer of 2013. His current research interests include hardware security and VLSI architecture for communications and signal processing.

Abhishek Basak received his B.E. (Hons.) from Jadavpur University, Kolkata, India in 2010 and is currently pursuing his Ph.D. degree in Computer Engineering at Case Western Reserve University, Cleveland, OH. He has many publications in premier conferences in the areas of biomedical system design for point-of-care health monitoring and hardware security, trust and validation. He is a student member of IEEE and EMBS.

Swarup Bhunia received his Ph.D. from Purdue University, USA in 2005. Currently, he is an T. and A. Schroeder associate professor of Electrical Engineering and Computer Science at Case Western Reserve University, USA. He has published over 150 articles in peer-reviewed journals and premier conferences. His research interests include low power and robust design, hardware security, and implantable electronics. Dr. Bhunia received National Science Foundation (NSF) career development award (2011), Semiconductor Research Corporation technical excellence award (2005), several best paper awards and SRC Inventor Recognition Award (2009). He is a senior member of IEEE.