A Built-In-Self-Test Scheme for Online Evaluation of Physical Unclonable Functions and True Random Number Generators

Siam U. Hussain, Mehrdad Majzoobi and Farinaz Koushanfar

Abstract—In the emerging era of Internet of Things (IoT) where various physical entities are spontaneously communicating with each other and sharing sensitive information, it is prohibitive to have a global entity for maintaining the security of the complex web against environmental variations and active attacks. Therefore, it is crucial that each entity has the capability of safeguarding its security features on its own. Methods based on harnessing the random identification and authentication from the physical device and environment, such as Physical Unclonable Functions (PUFs) and True Random Number Generators (TRNGs), if securely run, are promising primitives for protecting lightweight IoT devices. This paper presents the first Built-In-Self-Test scheme for on-the-fly evaluation of PUFs that can also be utilized for assessing the desired statistical properties of TRNGs. Unlike earlier known PUF evaluation suites that were software-based and offline, our methodology enables online assessment of the pertinent statistical and security properties all in hardware. Specifically, the BIST structure is designed to evaluate two main properties of PUFs: unpredictability and stability. Our work is the first online test suite that thoroughly evaluates the internal health of the entropy source of TRNGs along with the statistical properties of the generated bit stream. Comprehensive real-time evaluation by the BIST method is able to ensure robustness and security of both TRNG and PUF in the face of operational, structural, and environmental fluctuations due to variations, aging or adversarial acts. Proof-of-concept implementation of our BIST methodology in FPGA demonstrates its reasonable overhead, effectiveness, and practicality.

Index Terms—Built-In-Self-Test, BIST, Physical Unclonable Functions, PUF, True Random Number Generators, TRNG, Hardware security, Internet of things, IoT

1 Introduction

The Internet of Things (IoT) is a fast evolving paradigm where remotely identifiable physical objects form an active network, and are able to operate independently without external control. With these *smart* devices automatically connecting to each other and exchanging private and sensitive data, the potential for malicious attacks greatly increases. As a result, it is widely accepted that the most challenging aspect of IoT will be to ensure security and privacy. Moreover, due to potentially large scale of the network and dynamic communication among the devices, scalability is also expected to be a critical issue. This demands self-management, adaptability to situations and autonomic capabilities of the devices involved [1]. Therefore, a compact security solution which can maintain integrity and stability on its own is vital to ensure the integrity of an IoT system.

One of the primary properties of any IoT device is a 'unique identifier' [1]. A compact embedded identifier with a small footprint and energy budget is ideal for IoT as a large number of devices will operate in passive mode with limited source of power. Therefore, hardware-based security is well suited for IoT protocols and algorithms [2]. They are also naturally more resistant to physical and side-channel attacks. These facts make Physical Unclonable Functions (PUF) a promising enabler for generation of inherent and indelible device identifiers for IoT security solutions.

PUF is a (partially) disordered physical system: when interrogated by a challenge (or input, stimulus), it generates a unique device response (or output). The response depends on the incident challenge, specific physical disorder and PUF structure. It is common to call an input and its corresponding output a *challenge-response pair (CRP)*. PUFs have been established as an efficient embedded identification primitive over the last decade. Identifying and authenticating individual entities among a large variety of physical devices as demanded by IoT protocol is a challenge. Classic identification methods including serial numbers on-chip/package and ID storage in non-volatile memory are subject to attacks such as removal and remarking. Unclonable marking on the device ICs is important in that it can enable a low overhead identification, fingerprinting, and authentication for a wide range of disparate devices [3].

However, environmental factors, aging, or attacks from the adversary may deteriorate PUF performance and thereby cause delays, and more importantly, security threats. Therefore, they cannot guarantee a fully stand-alone solution as the root-of-trust of IoT entities without a careful analysis. As shown several times in practice, when

- *S. Hussain, and F. Koushanfar are with the Department of ECE, University of California, San Diego, CA 92093*
  Email: {s2hussai, fkoushanfar}@ucsd.edu
- *M. Majzoobi is with Mesh Motion Inc, Kensington, CA 94708*
  Email: m.majzoobi@gmail.com

Manuscript received 14 May 2015; revised 28 Oct. 2015; accepted 3 Jan. 2016. Date of publication 20 Jan. 2016; date of current version 8 Apr. 2016. Recommended for acceptance by S. Ray, J. Park, and S. Bhunia. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TMSCS.2016.2519902

the random PUF response values are not exactly random, catastrophic security failures occur. For example, several analysis and attacks on PUF have highlighted the need for appending input or output transformations for safeguarding purposes [4], [5], [6]. Moreover as devices connect and share automatically in IoT, stability of the response over a wide range of operational conditions is essential to ensure seamless operation.

To date, there have been very few papers on the evaluation of usability (stability or repeatability) and security (unpredictability or randomness) properties of PUFs. A comprehensive methodology to test the security of PUFs was first introduced in [7]. Their method was based on testing the randomness of the responses or conditional probability of responses (or a transformation of them) given the challenges, e.g., properties of the probability distributions. Latter research efforts, e.g., [8], [9], [10], have defined a more formal set of properties to evaluate PUFs. However, their primary focus was the evaluation of different PUF designs rather than real time monitoring of specific PUF instances. Moreover, to the best of our knowledge, all of the existing PUF assessment methods rely on software-based evaluation of outputs or CRPs.

This paper introduces the first methodology for online hardware-based assessment of the robust generation of streams of truly random (unpredictable) PUF responses that are unique to each device. Two main characteristics of PUF are evaluated by the Built-In-Self-Test (BIST) scheme: stability and unpredictability. These evaluations can reveal the operational, structural, and environmental fluctuations in the PUF behavior that may be caused by variations, aging, or attacks. To demonstrate this Built-In-Self-Test (BIST) scheme we analyze the PUF presented in [4].

The continuous and online monitoring of PUF characteristics yields several advantages including: (i) detection of changes/attacks during the PUF operation, (ii) providing an on-the-fly measure of confidence on the randomness/robustness of the CRPs, (iii) ensuring PUF stability by validating CRPs before adding them to library, (iv) reporting the exact conditions in the local PUF test site for a more granular debugging, and (v) enabling active adjustment and improvements of the PUF operations.

True Random Number Generator (TRNG) is another important cryptographic primitive that shares overlapping properties and often have structural similarities with PUFs. The integrity of any secure system depends largely on the proper functioning of these units. True Random numbers are needed for encryption keys, nonces, padding bits and many more applications. Software approaches to generate random numbers, called Pseudo Random Number Generators (PRNG) are widely adopted and are suitable for most applications. However, they are susceptible to attacks as their outputs are deterministic functions of the seed, which carries all its entropy. Therefore, hardware-based mechanism to extract randomness from physical phenomena is required for cryptographic applications. As data privacy is one of the key requirements of IoT framework, it can benefit from high entropy TRNGs generating secure keys.

A number of papers have focused on online evaluation of TRNGs [11], [12], [13]. However, a limitation common to all these work is that they deal only with the statistical properties of the output bit sequence of the TRNG. While these tests are important, complete security of the TRNG cannot be assured without evaluating its internal structure and the health of its entropy source [14]. BSIs standard AIS-31 [15] takes the construction of the TRNG into account during evaluation. NIST's Recommendation for the Entropy Sources Used for Random Bit Generation [16] also puts large emphasis on the quality of the entropy source and suggests several tests to evaluate it. Evaluating the internal health of TRNGs exclusively requires online tests, as offline tests would reveal the internal bit sequence which may facilitate modeling attacks.

To the best of our knowledge, two work have previously addressed the online evaluation of internal health of TRNGs. The work in [17] presented a method to measure jitter that help evaluate oscillator based TRNG. Moreover, [18] presented on-the-fly tests for evaluating non-ideal TRNGs where the generated internal bit sequences are not independent and identically distributed. We propose the first BIST scheme that evaluates both the internal health of the entropy source of an ideal TRNG as well as the statistical properties of its final output and is not limited to any specific design. As a demonstration, this scheme is used to evaluate two TRNG designs: metastability based TRNG presented in [19] and oscillator based TRNG presented in [20].

For the BIST scheme to be usable we have to make sure that the TRNGs and PUFs pass the tests in general and the BIST is able to detect when they deviate from this general behavior. Therefore, we propose improvements to some of these designs to make them pass these comprehensive tests. We also show how the BIST scheme can help make the calibration of the PUF automated. In addition to these, we present an scheme to safeguard the BIST result bits against malicious interceptions.

Main contributions of this work are as follows.

- First BIST scheme for comprehensive online assessment of TRNGs and PUFs (in hardware).
- Online evaluations of the internal structure of TRNGs as well as the quality of the entropy source, along with tests for the statistical soundness of the output sequence.
- Quantitative online analysis of both PUF stability and its security (unpredictability).
- A simple but effective scheme to safeguard the BIST results bit against malicious interceptions.
- Development of low-overhead control circuitry for saving the test results for further analysis.
- Evaluation of a number of TRNG and PUF designs along with suggestions for improvements so that they pass the extensive tests performed by the BIST scheme.
- Proof-of-concept FPGA implementation that demonstrates the effectiveness, practicability, and reasonable overhead of the proposed architecture. The source code is available at github.com/siamumar/BIST_PUF_TRNG.

Note that a preliminary version of this work was published in [21]. The present paper expands the scope of the BIST scheme in several ways. First, we add the evaluation of TRNGs. Since unpredictability is an important property for both TRNGs and PUFs, the resource usage is greatly optimized by designing a BIST scheme that shares resources.

Besides these, we also include tests for internal structure or TRNGs. Second, we redesigned a portion of the PUF stability test to make it part of the CRP generation process to generate stable CRPs. Third, we propose a simple scheme to protect the result bits form the BIST. Fourth, we run extensive tests on a number of TRNG and PUF designs and propose improvements to ensure that they have a high success rate in our comprehensive evaluation.

The rest of the paper is organized as follows. In the next section we outline the basic properties of TRNGs and PUFs along with their implementations and standard tests for their evaluation. In Section 3 we discuss the methodology to evaluate the desired properties of TRNGs and PUFs and architecture of the BIST scheme. In Section 4 provides the details of our implementation and summary of resource usage. In Section 5 the evaluation results for different TRNG and PUF designs are presented. The related literature is surveyed in Section 6. The paper concludes in Section 7.

2 Background

2.1 True Random Number Generator

True Random Numbers harness randomness from natural phenomena like thermal noise, shot noise, clock jitter, circuit metastability etc. The two primary components of a TRNG are the entropy source and a mechanisms to convert the entropy to random bit stream. Both components are equally important for generating unpredictable and statistically sound random bits. This work focuses on TRNGs implemented on FPGA. We first discuss a number of TRNG implementations, followed by the properties of a good TRNG.

2.1.1 Current TRNG Implementations

FPGA has been a popular platform for designing TRNGs because of its reconfigurability and fast time to market. Several approaches to generate random numbers in FPGA have been reported in the literature. We discuss a few examples here.

Sunar et al. introduced a TRNG that sourced randomness from the jitter produced by ring oscillators [22]. They combined the output of several identical ROs with an XOR gate based on the statistical assumption that the combined spectrum of all the ROs would be filled with jitter. This is the most widely explored method to extract entropy in FPGAs. One important improvement was proposed by Wold et al. in [20] where they inserted a D flip-flop after each ring to reduce the switching activity at the XOR input.

Cherkaoui et al. replaced the ROs in Sunar's model with self-timed rings that provide a number of jittery synchronized signals having the same period and a constant mean phase difference between them [23]. By adjusting the time lapse to the jitter magnitude of one stage, it can be ensured that the full spectrum will be filled by jitters without any statistical assumption.

In the method proposed by Hisashi et al. [24] randomness is produced by tying the S and R inputs of an SR latch together to force metastability. The routing between S and R is critical in this design in order to reduce bias.

In the TRNG proposed by Majzoobi et al. in [25] a flip-flop is forced to metastable state by equalizing the signal arrival times at clock and data inputs using programmable delay lines (PDL). The PDLs were realized by controlling the propagation delay inside LUTs. They introduced a feedback mechanism to adjust the delay to ensure metastability is sustained.

We evaluate the TRNGs proposed in [20] and [25] using our proposed BIST scheme.

2.1.2 Evaluation Criteria for TRNG

All the TRNG designs discussed in the previous section require combining a number of TRNG units to produce enough entropy. For example, in RO based TRNG, each unit comprises of one RO and one Flip-Flop to capture the bit. We refer to the bit sequence generated by each individual unit as the *internal bit sequence*. These sequences are XORed to generate the final output sequence, which we refer to as the *external bit sequence*. The evaluation criteria common to both TRNG and PRNG is the statistical quality of the external bit sequence. Several batteries of randomness test are available for this purpose as discussed in section 2.3. However, two additional features needs to be evaluated for TRNGs.

One important part of the security proof of TRNGs is that the internal bit sequences are independent and identically distributed (iid). However, due to increased power line noise, or intentionally injected noise the sequences produced by these units may become correlated which might in turn facilitate an attacker. To ensure that a TRNG is cryptographically secure the distribution of the bits generated by its composing units needs constant monitoring on their independence.

The second feature that needs to be evaluated is the level of entropy generated by each of the units. A fall in entropy level in any of the units is an indicator of an active attack or imminent failure of the TRNG. Most TRNGs employ post processing techniques to improve the statistical quality of the generated bit sequence. However, these techniques do not add to the entropy of the system. So the entropy level should be measured based on the raw data before any post processing is applied.

2.2 Physical Unclonable Function

The key idea behind Physical Unclonable Function (PUF) is exploitation of inherent and naturally occurring physical disorder (fingerprint) of the device as its unique signature, e.g., silicon manufacturing variations. A PUF is a function whose output (response) depends on both the applied input (challenge) and the unique physical properties of the hardware where it resides. The challenge and response together from one challenge-response pair (CRP).

There are two broad categories of PUFs: weak PUF and strong PUF [26]. Weak PUFs allow only a limited number of CRPs and it is assumed that its responses are not accessible by adversaries. On the other hand, strong PUF supports a large number of CRPs. This ensures that even if an adversary has access to a large subset of CRPs, it cannot predict the yet unknown ones. Note that although our BIST scheme is applicable to both weak and strong PUFs, without a loss of generality, this paper focuses on strong PUF testing. Since a weak PUF only differs with strong PUF in the number of possible responses, evaluation of weak PUF randomness is simpler since it is confined to the small space of responses.

2.2.1 Current PUF Implementations

A number of different realizations of the strong PUFs have been reported to date. In an *Arbiter PUF* [27], [28] two electrical pulses race simultaneously through two paths consisting of several stages. The exact paths are determined by the challenge vector. After the last stage, an arbiter, usually a latch or flip-flop determines which signal has arrived first and generates a binary response based on that. In [29] the responses from several Arbiter PUF stages were XORed to generate the PUF response. A *Lightweight Secure PUF* [4] has a similar structure, but the input challenge is passed through a complicated mapping to increase security against modeling attacks. *Ring Oscillator PUFs* exploits the specific and unique delay caused by fabrication variation of oscillators ([30]). We evaluate an improved version of the lightweight secure PUF presented in [4] with our BIST scheme. To implement the FPGA PUF we use the comprehensive methodology introduced in [19], [31].

2.2.2 Evaluation Criteria for PUF

To make sure that a PUF is secure and reliable, it must possess the following properties [7]:

- *Unpredictability:* This property ensures the uniqueness of the PUF behavior, i.e., the CRPs. For the PUF to be truly unpredictable two conditions need to be met: First, the response bits from the PUF should be completely random. Second, the transition of the response bit should be completely uncorrelated with the transition of one or more bits of the challenge vector. The second condition ensures that the response from the PUF cannot be predicted based on known challenge-response pairs.
- *Stability:* In a strict sense, for a PUF to be used as an identification circuitry it must always generate the same response when excited by the same challenge vector. Since PUF uses physical components and are inherently noisy, this criterion is difficult to meet precisely. But the stability shall be present in a majority of the output bits to ensure usability.

One question that may arise is the need for online testing of PUFs. One may argue that PUFs can be evaluated offline for their randomness property during the initial testing phase; since CRPs need to be stable, there is no need for further online tests. This argument has at least two ramifications: (i) since the CRPs have to be robust, the protocols have to work even if the response bits change up to a certain percentage (typically 20% in current generations of PUFs.) One could seriously bias the bits within this high percentage of robustness (e.g., by changing the temperature) to facilitate machine learning attacks [5]; (ii) one may use the side-channel information at various operational points to break the PUF security as demonstrated by the work in [6], [32]. Adapting the randomness tests for online PUF evaluations would detect these scenarios and could even provide a sensing mechanism to avoid such attacks and thus, improve the robustness of PUFs.

For more information about the PUF, its properties, and recent directions, we refer the interested readers to comprehensive articles on this topic [26], [33].

2.3 Standard Randomness Tests

Several standard test suites such as NIST [34], DIEHARD [35], FIPS [36] are available for evaluating the statistical quality of the bit sequence generated by RNGs. NIST published recommendations for the design of entropy sources and the tests for the validation of these sources [16]. They also published documents with recommendations for deterministic algorithms to produce pseudorandom values from an entropy source [37] and on how to combine these two mechanisms to generate a good quality TRNG [38]. BSI (Bundesamt fr Sicherheit in der Informationstechnik), also established evaluation standards for both PRNG (AIS-20 [39]) and TRNG (AIS-31 [15]).

The batteries of randomness tests were originally designed to evaluate the performance of the RNGs. As discussed above, randomness of the response bits of PUF is mandatory to ensure that the PUF cannot be modeled as a deterministic process. Here, besides evaluating TRNGs, we utilize the standard randomness test suite to evaluate the unpredictability of PUF response.

3 Architecture of the BIST Scheme

The BIST scheme for both TRNG and PUF are developed such that they can be used either independently or together, in which case they can share a large portion of the resources. For the sake of clarity, we describe the architecture of the two schemes separately. The exact implementation details of the BIST is outlined in Section 4.

We would like to mention that this BIST scheme is for real-time monitoring of TRNG and PUF instances during their operation. Initially, any new PUF or TRNG design have to undergo rigorous test to verify their effectiveness and security. Many of these tests are too complex to be performed in real-time. Moreover, as already explained, their primary purpose is to evaluate designs not an individual instance of PUF or TRNG. However, even if a design passes these initial evaluations, these structures are unstable and unpredictable by nature and may exhibit short time dip in performance for inexplicable reasons. Active attacks may also create security threats. The purpose of the proposed BIST scheme is to maintain the security of the system in such scenarios.

3.1 Architecture of TRNG Evaluation Scheme

As mentioned in Section 2, most TRNG implementations employ multiple units and take XOR of the generated sequences to get a good quality random output. While it is important to monitor the quality of the final output, monitoring the health of the internal units is even more important to ensure the integrity of the TRNG and to predict a possible failure. Therefore, we perform tests on both internal and external bit sequences. Fig. 1 shows an overview of the architecture of the TRNG evaluation scheme.

3.1.1 Test on External Bit Sequence

This test evaluates the quality of the final output sequence (after XOR operation). It incorporates seven tests from the NIST battery of randomness test [34]:

1) Frequency
2) Block Frequency

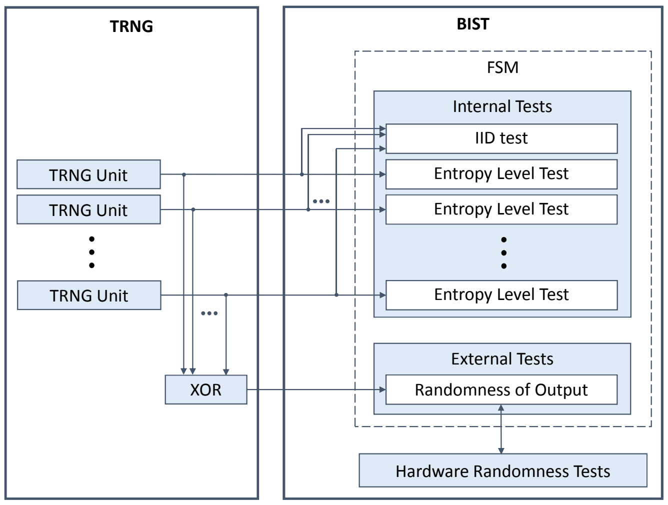


Fig. 1: Architecture of the TRNG Evaluation Scheme

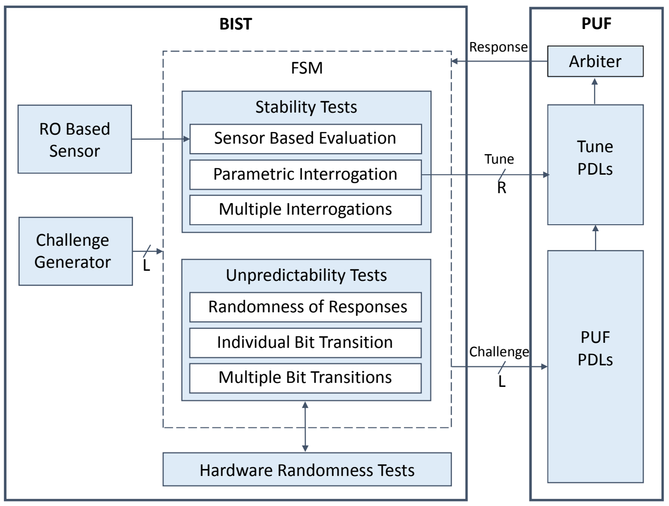


Fig. 2: Architecture of the PUF Evaluation Scheme

3) Runs
4) Longest Run of Ones
5) Non-overlapping Template Matching
6) Overlapping Template Matching
7) Cumulative Sums

The test module resides on the same chip as the TRNG. Each test operates on a block of random bits and outputs one for success and zero for failure.

3.1.2 Test on Internal Bit Sequence

We perform test on internal bit sequences to check
(I) if the bit sequences from each unit are independent and identically distributed (iid)
(II) the level of entropy generated by each unit.

IID Test

The NIST recommended test for iid in [16] requires complex calculation and most importantly a large amount of data to be stored on chip. While this test is essential at the initial evaluation of the TRNG design, it is not suitable for online test. We utilize the Wald-Wolfowitz Runs test [40] to determine whether the bit sequences produced by different TRNG units are iid or not. Unlike the Runs test in [34] which needs the Frequency test to be passed as a pre-condition, the Wald-Wolfowitz Runs test does not put any condition on the frequency of '1' or '0' bits. Internal sequences almost always fail the Frequency test (verified by offline testing).
An error signal is generated if the iid test fails.

Entropy Level Test

We employ a simplified version of the procedure described in [16] to estimate the level of entropy generated by each of the TRNG units. A warning signal is generated if the entropy level drops below a certain threshold in one or two of the units and an error signal is generated if that happens in more than two units.

3.2 Architecture of PUF Evaluation Scheme

The PUF Evaluation scheme is much more complex than that for TRNG. It evaluates the PUF under test based on the two major properties described earlier: unpredictability and stability. The overall architecture of scheme is shown in Fig. 2. The finite state machine (FSM) controls the aforementioned two test sets, each of which consists of three tests that shall be described in Sections 3.2.1 and 3.2.2 respectively. The BIST generates the challenges as appropriate for each test. In addition to outputting L-bit challenges to the PUF input, the BIST output also includes the appropriate R-bit PUF tuning parameter. The input to the BIST block is the PUF response. The randomness tests are performed by the block denoted as "hardware randomness tests" which is shared with the TRNG evaluation scheme. There is also a random challenge generation component.

3.2.1 Unpredictability Test

The unpredictability tests check the randomness of the response from the PUF. It uses the same randomness test module that is used to test the TRNG as described in Section 3.1.1. Unpredictability is evaluated with the three following tests, denoted by UT1, UT2, and UT3 respectively.

UT1: Randomness of Response(s)

In each round of this test, the PUF is excited with a set of random challenges and the corresponding response bits are fed to the NIST module. The test is repeated for several hundred rounds and the result bits are added up to calculate the average success rate over the multiple test rounds.

UT2: Effect of Individual Bit Transition

This test checks whether an output bit transition is correlated with the transition of any particular challenge bits. First, the PUF is excited with an L-bit random challenge vector. The PUF is then interrogated by the same vector with only the i-th bit inverted $i \in \{1, 2, \dots, L\}$. The XOR of the response bits for these two challenges (differing in one bit) are applied to the test module. For an unpredictable PUF, the transition in the response must be completely arbitrary (i.e., uncorrelated with the input bit change). Thus, the XOR of the two response bits must pass the randomness tests. This test is performed several hundred times for each value of i ($i = 1, 2, \dots, L$) and the success rate is reported as a function of the inverted bit index, i.

UT3: Effect of Multiple Bit Transitions

This test checks for the correlation between the response bit transition and transition of one or more challenge bits. First the PUF is excited with an L-bit random challenge vector. The PUF is then interrogated by a similar vector where h bits are inverted. Thus, the Hamming distance between these two challenge vectors is h, where $h \in \{1, 2, \ldots, L\}$. The XOR of the responses in these two cases is fed to the randomness test module. This test is performed several hundred times for each value of h and the success rate is reported as a function of the Hamming distance, h.

3.2.2 Stability Tests

The PUF operation is very sensitive to the silicon, environmental and operational conditions. One way to ensure stability is to comprehend these conditions and their impact of the responses, and then stratify the results for each situation. Hardware-based sensors are very useful for this purpose and they form our first stability testing method.

The second proposed methodology for stability testing utilizes multiple evaluations of the same input. A simple consensus method can be used for combining the results of multiple tests and enhance the stability. The third and last stability testing method is based on changing the parameter(s) affecting the stability and then check the impact on the response.

ST1: Sensor-based Evaluation

The most impactful conditions on the PUF operation are those that change the randomness (entropy) of the circuit output. Some of the conditions that could affect the PUF behavior include: light, magnetic field, IC aging, and on-chip voltage and temperature. A good BIST for PUF shall include sensors that comprehend these conditions. Perhaps the simplest and most widely suggested/used sensors for on-chip sensing of the process variation, temperature, and aging are the architectures based on ring-oscillators (ROs). In particular, the RO-based sensors exploits the temperature dependence of the threshold voltage and carrier mobility of CMOS transistors that in turn affect the frequency of an RO.

ST2: Multiple Interrogations

To test whether the PUF is able to reproduce the same response, the multiple interrogation method excites the PUF with the same challenge vector several times and computes a consensus of the response bits to the repeated challenges. This simple method was suggested and used in [31] for lowering the error in the responses. The FPGA prototype of this method demonstrated that even a small number of repetitions could improve the response stability as the width of the transition region gets narrower and more statistics is gathered.

ST3: Active Parametric Interrogation

The PUF randomness is typically converted to a digital binary format using a metastable component, for e.g., an arbiter. A set of parameters, in combination with the challenge vector, drive the point of operation of this metastable part. For certain challenges, small change in this random parameter may potentially alter its operating point and change the corresponding response. In parametric interrogation, this parameter is altered in deterministic steps around its expected value and the resulting response is observed.

In the parametric interrogation for arbiter-based PUF suggested in [31], a programmable delay line is used to change the delay difference between the two arbiter inputs. If the response remains unchanged at both a positive and a negative delay differences, the response is stable and the incident challenge is marked as a *robust challenge*. Otherwise, the challenge is advised to be removed from the CRP library.

3.3 Protecting Result Bits

A possible location of attack is the output bits from the BIST scheme. If an attacker can intercept one or more of the output bits and force them to output '1', the user will not be able to detect any attack on the PUF or TRNG. To protect the BIST from such attacks we provide a 16 bit checksum which incorporates the output bits from all the unpredictability tests and IID test (statistical tests that provide *pass/fail* results). The user can form the checksum from the external output bits and periodically match it with the checksum provided by the BIST to verify the integrity of the output bits. We assume that an attacker is only able to force the output bits to a certain constant (in this case '1'). Since the other tests like Entropy level test provide quantitative results, such protection is not required for them.

4 BIST System Description

In Figures 1 and 2 in Section 3 the architecture of TRNG and PUF evaluation schemes are depicted separately for the sake of clarity. In the actual implementation, the hardware randomness test module, which constitutes a large portion of the BIST, is shared between the two schemes. In our design, the TRNG is monitored continuously while the PUF evaluation is done periodically. The TRNG evaluation is paused during the unpredictability test of PUF.

Along with the implementation of the BIST, we also describe the TRNGs and PUF under test for better understanding of the scheme and to make sense of the overhead estimation. We evaluated two TRNGs: the one based on ROs [20], [22] and the one based on metastability [25]. The PUF under test is a delay-based strong PUF proposed in [19]. It also incorporates the input and interconnect networks introduced by the lightweight secure PUF in [4]. The output network is just an XOR mapping with a Q-bit input and a 1-bit output.

The BIST module resides on the same chip as the PUF and/or TRNG and is able to run its tests automatically. At the end of this section, we report the resource utilized by each of the modules in Table 1.

4.1 TRNG Implementation

4.1.1 RO based TRNG

We evaluate the RO based TRNG proposed by Wold et al. in [20]. In this TRNG, randomness is extracted from the jitter present in the oscillating pulse of an RO. The pulse from each RO is sampled with a flip-flop to generate the bit stream. We used 16 ROs each comprising of 3 inverters. The bit streams from the 16 units are combined with a 5 stage XOR network to produce the final output.

4.1.2 Metastability based TRNG

In this TRNG, a flip-flop is forced to metastable state by equalizing the signal arrival times at the clock and D inputs. The propagation path to these inputs is controlled by a feedback system through Programmable Delay Lines (PDL). A PDL [19], [31] is composed of LUTs where the logical output depends on only one of the inputs and other inputs act as *don't cares*. These *don't care* inputs control the signal propagation path, and consequently, the delay through the LUTs.

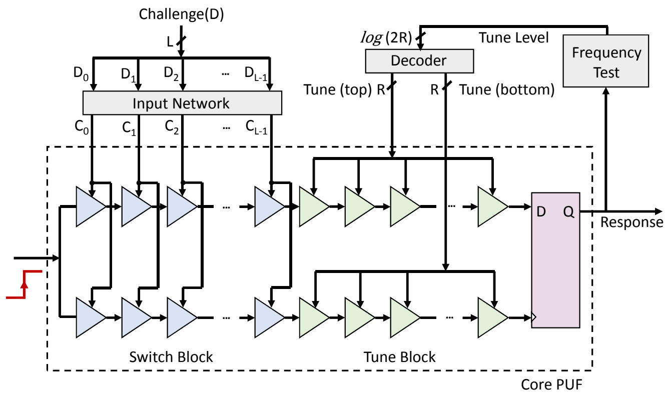
Operation of this TRNG depends on the delicate balance between the signal arrival times at clock and data input, which makes it very sensitive to the routing within the PDLs (see Section 2.1.1). As the peripheral circuitry becomes more complex, they affect the routing and at some point the routing of the two lines become so different that the feedback system is not able to compensate for that and the TRNG starts producing constants. To solve this issue, we implemented hard macros for the PDLs and instantiated the same hard macro for both PDLs. In this way, they are exactly symmetric except for the routing of clock to input and output to the flip-flop. Besides solving the issue of the peripheral circuit effect, it reduces the required number of LUTs in each PDL to only 8 instead of 64 in the original design. The original design employed only one TRNG unit and it required complex post-processing to ensure that the generated sequence passes the statistical tests. As our implementation requires fewer LUTs, we were able to have several units and XOR their outputs to have a good quality random sequence.

4.2 PUF Implementation

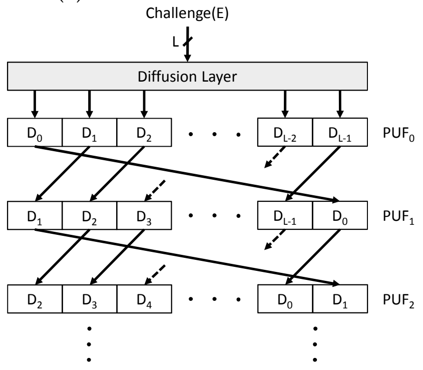
We evaluate a delay based PUF using the BIST scheme. The principles of the delay based PUF were first introduced by Gassend et al. in [28]. In this PUF, generating one output bit requires a step signal to travel through two parallel paths composed of multiple segments that are connected by a series of 2-input/ 2-output switches. Each switch has a selector bit which decides whether it is configured as a cross or straight connector. The path segments are designed to have the same nominal delays, but their actual timings differ slightly due to manufacturing process variations. The difference between the top and bottom path delays are compared by an arbiter that generates the response accordingly (e.g., response is '1' if top path is faster). The PUF challenges act as the selector bits of the switches.

In our implementation, the PUF structure has two main components: the core PUF unit presented in [19] and the peripheral circuitry to enhance PUF security presented in [4]. Moreover, we propose an automated calibration of the PUF using the BIST scheme and also propose improvement to the peripheral circuitry to make sure the PUF passes the comprehensive tests by the BIST.

The PUF circuit is shown in Fig. 3. To increase the unpredictability (randomness) in the PUF response and make it more secure, responses from $Q(=16)$ parallel PUFs are XORed (similar to TRNG structure). We refer to each parallel PUF (before XOR) as one PUF unit. which is shown in Fig. 3a. The arrangement of these units is shown in Fig. 3b



(a) Structure of one PUF unit


Challenge(E)
L
Diffusion Layer
D_0 D_1 D_2 ... D_{L-2} D_{L-1} PUF_0
D_1 D_2 D_3 ... D_{L-1} D_0 PUF_1
D_2 D_3 D_4 ... D_0 D_1 PUF_2

(b) Arrangement of the parallel PUF units

Fig. 3: Structure of the PDL based PUF

4.2.1 Core Unit of PUF

The switches of the delay based PUF are realized by the programmable delay lines (PDL) which are described in Section 4.1.2 (The triangular elements represent the LUTs of PDL in Figure 3a). The path with a larger delay is equivalent to the cross connector in the switch and the the path with shorter delay is equivalent to the straight connector. As in the implementation of the TRNG, the PDLs are designed as hard macros to minimize bias due to routing asymmetry.

PDLs are also used in the tuning block that minimizes the bias in path delays. The tuning block inserts extra delays into either the top or bottom path based on their selector inputs to compensate for the delay bias caused by the routing asymmetry. The selectors of the top and bottom PDLs in each tuning block are controlled independently while a single selector bit drives both corresponding LUTs in each PDL in the switch block.

Each of the Q PUF units consists of L switches and R tuning blocks. In our implementation, we use $L = 64$, $R = 16$ and $Q = 16$.

4.2.2 Calibration

The PUF goes through a calibration phase at the start of its operation where optimum tune levels for each PUF unit are set to minimize the delay bias. In [19], a set of 64000 CRPs were collected for each PUF unit at each tune level. Then the tune level at which the mean of the responses were closest to half were chosen as the optimum one for that unit. However, the collection of so many CRPs ($64000 \times R \times Q$) make the calibration process tedious and time consuming. Moreover, it would require reading the response from each PUF unit while during regular operation only the XORed response is read.

Since the BIST scheme holds a random challenge generator (Section 4.3.4) for performing the tests, we also use this for generating the challenges for the calibration process. The responses from each PUF unit are read inside the FSM (see Section 4.3.7) and it sets the tune level automatically following the process described above. The decoder then sets tune (top) and tune (bottom) such that their Hamming distance is equal to the tune level.

4.2.3 Peripheral Circuitry

The peripheral circuitry adds security to the PUF by using input/output transformations that thwart active attacks. Its first three components were part of the original design in [4]. This design passes the first unpredictability test (UT1) described in Section 3.2.1. However, the success rates for UT2 and UT3 were close to zero. To solve this, we added a diffusion layer in front of the interconnect network.

1) Input Network: The input network transforms the challenge by a function satisfying the Strict Avalanche Criterion (SAC). A function is said to satisfy SAC if, whenever a single input bit is complemented, each of the output bits changes with a probability of 0.5. The transformation used in our implementation is:

$$c_{(N+i+1)/2} = d_i, \; for\, i = 1 \tag{1}$$

$$c_{(i+1)/2} = d_i \oplus d_{i+1}, \; for\, i = 1, 3, 5, \ldots, N-1 \tag{2}$$

$$c_{(N+i+2)/2} = d_i \oplus d_{i+1}, \; for\, i = 2, 4, 6, \ldots, N-2, \tag{3}$$

where d is the input to the input network and c is the output that is fed to the PUF.

2) Output Network: In our implementation, the output network is a Q-bit XOR gate implemented by LUTs. As explained in [41] XOR operation reduces the bias present in a set of random sequences if they are independent and uncorrelated.

3) Interconnect Network: If an adversary knows the structure of the input network, she would be able to compute its inverse and pass the challenge through it to nullify the security effect of the input network. To thwart this the challenge is transformed through the interconnect network as shown in Fig. 3b. With this network present, the inverse of each of the Q input networks is distinct and it is not possible to bypass more than one of them.

4) Diffusion Layer: The purpose of the diffusion layer is to make sure that the probability of flip in the response bit is close to half when one or more bits at any position in the challenge vector is flipped. The second and third unpredictability tests in Section 3.2.1 check for correlations of the flip in response bit with flips in a single and multiple challenge bits respectively. With only the above three elements of the peripheral circuitry which were proposed in [4], the success rate for these two tests are close to zero. The possible reasons are either (a) the effect of few

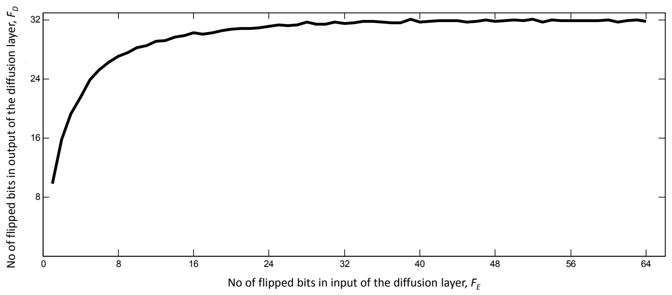


Fig. 4: Average no of flipped bits in output, F_D vs no of flipped bits in input, F_E of diffusion layer for input length, $L = 64$.

bits on the response bit is not enough, so it the response never flips or (b) the effect is very strong, so it always flips.

As a remedy to both of the above causes, the diffusion layer distributes the effect of a flip in a single bit in its input, E over multiple bits in output, D. This layer is essentially a multiplication by a constant matrix similar to AES. The 64-bit challenge vector is divided into sixteen 4-bit numbers and these sixteen numbers are arranged as a 4×4 matrix. This matrix is then multiplied by another 4×4 constant 2-bit matrix with rank (number of independent rows) of 3. Higher rank of the matrix results in more diffusion. However, to ensure that this diffusion layer cannot be bypassed by placing its inverse matrix in front, we chose a matrix with determinant of zero. Therefore, the highest possible rank is 3.

Fig. 4 shows the relation between the number of flipped bits in output, F_E and number of flipped bits in input, F_D for input length, $L = 64$. For $F_E = 1, F_D \approx 9$. Initially F_D increases with increase in F_E. However, as F_E increases more, F_D asymptotically reaches 32 ($= L/2$). This trend works against both the causes mentioned before and makes sure the PUF passes UT2 and UT3.

In our implementation platform, Virtex 6, the delay through the diffusion layer is about $4ns$ while delay through the core PUF is about $160ns$. Therefore, the delay caused by the diffusion layer is negligible.

4.3 BIST Scheme

The BIST architecture incorporates a randomness testing module, iid test module, entropy estimation module, a challenge generator, an RO based sensor, and a Finite State Machine (FSM) which controls the flow of all the tests.

4.3.1 Randomness Test Module

The randomness test module incorporates seven tests from the NIST test suite [34] listed in Section 3.1.1. These tests involve complex mathematical functions like *complementary error function (erfc)* and *incomplete gamma function (igamc)* that are unsuitable for hardware implementation. In our implementation, we adopt the simplifications suggested in

[12]. Equations (4) and (5) show the general form of simplifications on the *erfc* and *igamc* functions respectively.

$$P\_value = erfc(x)$$
$$P\_value > \alpha => x < erfcinv(\alpha) \quad (4)$$
$$P\_value = igamc(a, x)$$
$$P\_value > \alpha => x < igamcinv(a, (1-\alpha)) \quad (5)$$

Here, x is a function of a counter value and the input sequence length n; a is a function of the number of blocks N; and α is the Type I error probability. For a constant set of n, N and α, one can use the above equations to set conditions on the counter value. The counter accumulates the respective test metric. For details of each test implementations please refer to [12].

In [34], the Frequency test is noted as a prerequisite to the Runs test and [12] utilizes this fact to simplify the Runs test implementation. Therefore, we do not implement the Frequency test separately. Note that a recent work in [13] presents more optimization by sharing resources among these tests. This scheme can be seamlessly incorporated in our design.

4.3.2 *IID Test*

We implement the Wald-Wolfowitz Runs test to evaluate the independence of the distribution of bit sequences produced by different TRNG units. It requires three counters to count the number of bits, number of ones, and number of runs, one multiplier which is time shared for computations of the mean and variance of the distribution and two comparators to generate the final result. The test sequence is generated by cyclically taking one bit from each TRNG unit. If there are Q units and the bit produced by the i-th unit at cycle c is $b_i(c)$, the test sequence is
$[b_0(c), b_1(c), \ldots, b_Q(c), b_0(c+Q), b_1(c+Q), \ldots]$.
Since Q internal bits are generated in each clock cycle, the Q-bit vector is read in cycle c and the vectors for the next Q clock cycles are ignored so that this test runs at the same clock as other parts of the design. The runs test depends on the relation between the consecutive bits in the test sequence. The inconsistency in filling this sequence takes place only once in Q bits while all other consecutive bits are generated at the same instance. Moreover, the probability that distribution will change within this time period should be very low. Therefore, the resultant error should be negligible.

4.3.3 *Entropy Estimation*

To estimate the level of entropy we follow the procedure provided in [16]. The equation for lower level of entropy bound, E is given by,

$$E = \min\{W, -\log_2(\frac{1}{N}(C_{max} + 2.3\sqrt{C_{max}(1-\frac{C_{max}}{N}}))))\} \quad (6)$$

where the test sequence is generated by grouping W bits of the input sequence, $C_{max}$is the number of occurrence of the most common element in the test sequence, and N is the length of the test sequence.

Precise calculation of entropy would require operations like square root or logarithm which are costly in resource and time. While this is important during initial evaluation of an entropy source, for an online test estimating the level of entropy up to certain precision is sufficient. Our implementation provides the entropy level of each TRNG unit in a scale of 0 to 7. Figure 5 shows the entropy as function of C_{max} for for $W = 2$ and $N = 10000$. The minimum entropy associated with each level is shown in the left y-axis.

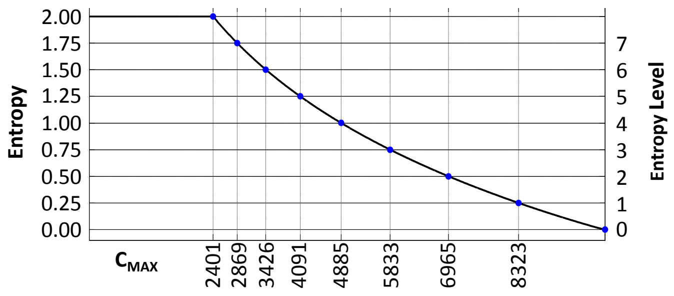


Fig. 5: Entropy level vs C_{max} for $W = 2$ and $N = 10000$

Implementation of this simplified entropy estimation scheme requires four counters to count the occurrence of the four 2-bit words (for $W = 2$), six comparators to determine C_{max} and eight comparators to determine the entropy level. There need to be one estimator for each of the TRNG units. In our implementation, entropy estimation of the i-th unit is started at i-th clock cycle so that the two comparators are shared between the entropy estimators for all the units. Only the resource used by the counters is proportional to the number of TRNG units, Q.

4.3.4 *Challenge Generator*

To perform one round of tests described in Sections 3.2.1 and 3.2.2, several gigabytes of random challenge vectors are required. To generate these challenge vectors, we implement an on-chip true random number generator (TRNG). We adopt the implementation suggested by Wold et al. in [20]. It generates random numbers at a 132 Mbits/sec rate. In this particular implementation, generating a single bit response requires a 64-bit challenge. We concatenate the outputs from four TRNGs to generate one challenge vector.

4.3.5 *RO Based Sensor*

Zick et al. presented a compact RO based sensor for online measurements of variations in delay [42]. We incorporate that design in the BIST-PUF to monitor the stability of the delay-based PUF. This sensor consists of an RO and a frequency counter.

4.3.6 *Parametric and Multiple Interrogation Schemes*

In actual implementation, this scheme resides inside the PUF module. It needs two additional signals: an input signal, GEN/AUTH which indicates if the challenge is applied for CRP generation or authentication and an output signal RESP_VALID which indicates if the applied challenge passed ST3: Active Parametric Interrogation (Section 3.2.2). During CRP generation the challenge is applied V times at each of the following three settings of the tune level of the PUF: (i) extra delay on top path, (ii) extra delay on bottom path, and (iii) similar delays. The response from each excitation is summed up. If the final sum is close to either $3V$ or 0 the RESP_VALID signal is set to '1' which indicates that challenge is robust. Otherwise, it is set to '0' which indicates to the user to mark the challenge as unstable and discard it from the CRP table. During authentication, only the robust challenges are applied and the tune levels are set to its optimal condition required to ensure unpredictability.

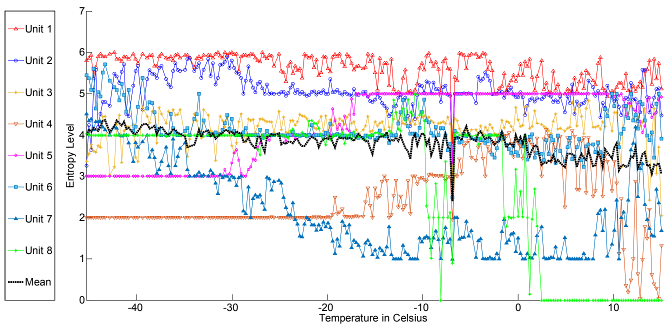
We employ two multiple interrogation schemes: one is incorporated in the PUF structure to generate the response by voting and the other is part of the BIST scheme to assess the stability of the PUF. In the first one, the challenge is applied T_1 times and the responses from each excitation are summed. The *final* response is set as the MSB of the sum, which means the response is '1' if $sum > (T_1 - 1)/2$, and '0' otherwise. As we show in the evaluation, PUF stability is greatly enhanced by employing both parametric and multiple interrogation schemes during CRP generation. The second multiple interrogation scheme, which is part of the BIST, sums up these *final* responses for T_2 times. If this sum S_R is close to either T_2 or 0 the response is stable. On the other hand if S_R is close to $T_2/2$, the response is unstable.

4.3.7 FSM

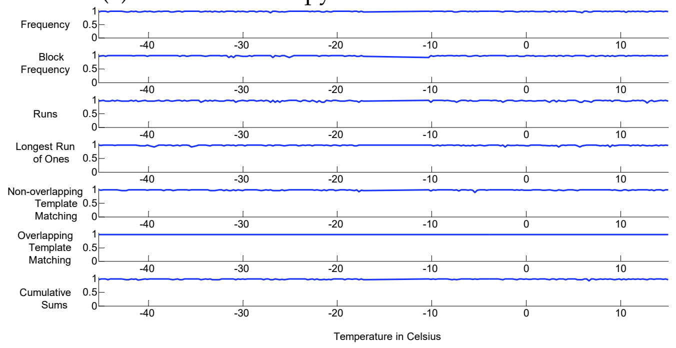
The FSM is designed to control the tests described in Sec 3. The tests for TRNG evaluation are relatively simpler. The FSM reads the test results and sums them up for several hundred instances and reports the mean success rate. Our design provides both instantaneous results for real-time monitoring and cumulative results over several hundred rounds of testing. The cumulative results are stored in a memory block and can be later read to have a full assessment of the TRNG performance using a more comprehensive data. The BIST scheme also sets up error/ warning signals when the success rate goes below a certain level.

The PUF requires complex control elements to perform complete evaluation. The unpredictability tests require the randomness test module, which it shares with the TRNG evaluation scheme. To optimize the area, we designed the FSM such that it performs the three unpredictability tests on the PUF sequentially using the same module. The random challenge vectors from the challenge generator are read by the FSM which performs the necessary manipulations on them as required by the tests UT2 and UT3 (Section 3.2.1). The responses from the PUF are sent to the test module via the FSM. Similar to TRNG evaluation scheme, both cumulative and instantaneous results are available. The FSM also generates the checksum from the output bits that provide instantaneous results as explained in Section 3.3.

Our sensor based evaluation reads the delay measurement data from the RO-sensor and saves it for future reference. For the multiple interrogation test, the tuning parameters are set to the optimum values. The same challenge is applied 100 times and the sum of the responses are stored in the memory. A sum close to either 0 or 100 indicates a stable PUF. This test is done entirely inside the FSM without using any external module. The Active Parametric Interrogation test is performed inside the PUF module as explained in the previous section.



(a) Estimated entropy level of different units



(b) Mean success rate of the output sequence in NIST tests

Fig. 6: Evaluation result for metastability based TRNG

4.4 Overhead Estimation

We implement the BIST scheme on a Xilinx Virtex 6 FPGA. The resource used by different components of the BIST scheme is reported in Table 1. These numbers are functions of different parameters, e.g., number of inverters/PDLs in the TRNG, the test sequence length, block size or maximum latency in the test module. Therefore, the parameters used in our implementation are also reported.

As mentioned in 3 the randomness test module is required to test both PUF and TRNG. IID test and entropy estimator is used to test only TRNG and and challenge generator and RO sensor is used only for PUF. All these components together use only 2% of the total resource available in Virtex 6.

We also report the estimated power usage of each component. The power is estimated with XPower Analyzer [43] from Xilinx. Note that the reported power estimation is for the case when the BIST is running. However, the BIST need not run continuously. Instead, it can be run at certain intervals to ensure safe operation.

5 Evaluation

We evaluated two TRNG designs, namely the TRNG based on ring oscillators [20] and TRNG based on metastability [25] and one PUF design namely the PUF based on programmable delay lines [19]. Here we present selected evaluation results. Type I error rate, α is set to 0.05 for all the statistical tests.

TABLE 1: Overhead Estimation

Component	Register/ LUT	Latency*	Power(mw)	Parameters
RO based TRNG	36/ 123		8.50	P = 3, Q = 16
PDL based TRNG	104/ 560		40.42	P = 8, Q = 8
PUF with Peripheral Circuitry	28/ 3454		175.66	L = 64, R = 16, Q = 16
Randomness Test Module				
1) Frequency	-	1	-	n = 20000
2) Block Frequency	32/ 50	2	1.13	N = 100, M = 200
3) Runs	64/ 138	2	0.17	n = 20000
4) Longest Run of Ones	149/ 307	2	0.79	N = 16, M = 8
5) N.O. Template Matching	30/ 35	2	0.63	N = 8, M = 256
6) Overlapping Template Matching	273/ 268	68	3.38	N = 1000, M= 1023
7) Cumulative Sums	77/ 136	2	1.80	n = 20000
IID Test	71/ 63	3	20.00	n = 16384
Entropy Estimator	713/ 1193	3Q (= 24)	40.01	W = 2, N = 10000, Q = 8
Challenge Generator	144/ 492		6.14	
RO-Sensor	8/ 14		0.00	
FSM	182/ 618		37.58	

P: No. of elements (inverters/PDLs) in each TRNG unit
Q: No. of parallel units in TRNG/PUF
L: Length of the challenge vector o PUF
R: Length of the tune block of PUF

n: Length of each test sequence
N: No. of blocks in each test sequence (if applicable)
M: Length of each of each block (if applicable)
n = MxN (if applicable)
W: Word size

*Maximum no of clock cycles between the last bit of the test sequence is generated and the result is valid

5.1 Evaluation of TRNGs

5.1.1 Evaluation of Metastability Based TRNG

The estimated entropy level of different units and mean success rate for the seven NIST tests at different temperatures are plotted in Figure 6 (a) and (b) receptively. The mean entropy of all the units is also shown. Even though the entropy level of the units fluctuates widely, the mean entropy level follows that of unit 1 as its value is higher than the others. At around $-6°C$ there is a dip in the entropy level which also manifests itself in the success rates of the NIST tests. As mentioned in Section 4.3.7 each test is performed 250 times and the mean success rate is reported. Moreover, each test is performed on a sequence of several thousand bits. Therefore, the dip in the curves shows that the TRNG produced low quality output for a long sequence.

One important fact to notice is that while the entropy level of most of the units either remains similar or increases (units 4 and 5) with temperature, the entropy level of units 7 and 8 decreases, which is contrary to the general concept that entropy should increase with temperature. This fact validates the importance of online tests as it shows that theoretical concepts do not always hold during TRNG operation.

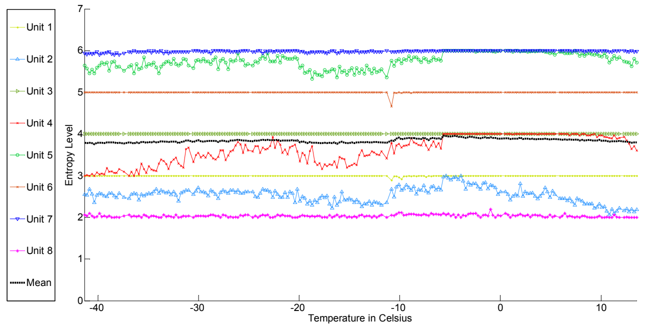
Note that the reported results are selected from many observations. For most of the cases the observations conform to the theoretical concepts. However, we present these particular observations to demonstrate that TRNGs may exhibit inexplicable behavior at times due to their physical nature and therefore real-time monitoring of its performance is crucial to ensure proper operation.

5.1.2 Evaluation of RO Based TRNG

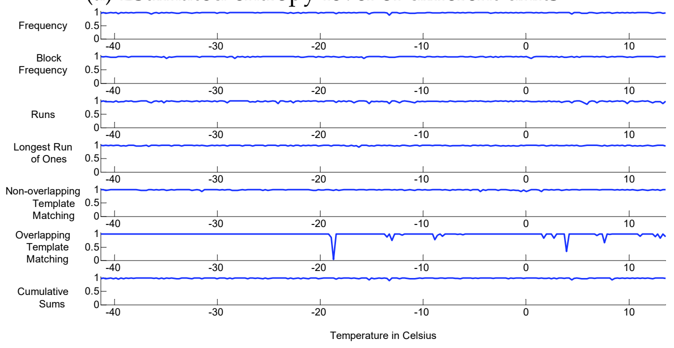
The evaluation results for RO based TRNG are shown in figure 7. Note that to improve readability, entropy level of only half of the units are plotted. The entropy levels remain relatively more stable as compared to the previous one. The entropy levels either remain similar or increase with temperature.

5.2 Evaluation of Unpredictability of PUF

The results of the three unpredictability tests described in section 3.2.1 are shown in Figure 8 for two different



(a) Estimated entropy level of different units



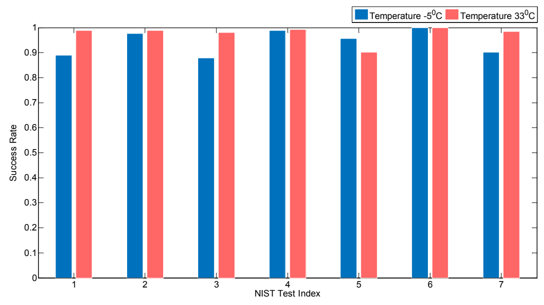
(b) Mean success rate of the output sequence in NIST tests

Fig. 7: Evaluation result for RO based TRNG

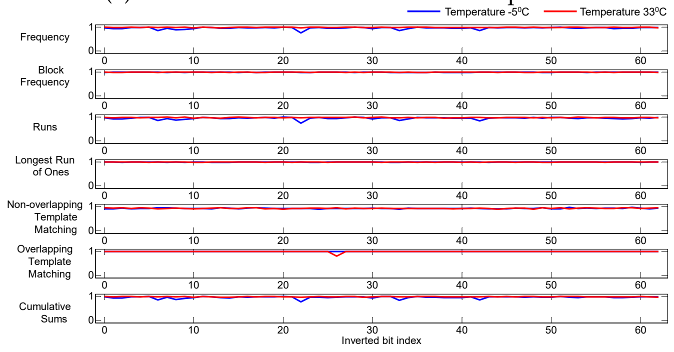
temperatures: $-5°$C and $33°$C. Although the success rates for these tests are reduced slightly at lower temperature, the difference is negligible. Note that the original design presented in [4] does not include the diffusion layer without which the PUF passes only UT1 but fails UT2 and UT3.

5.3 Evaluation of Stability of PUF

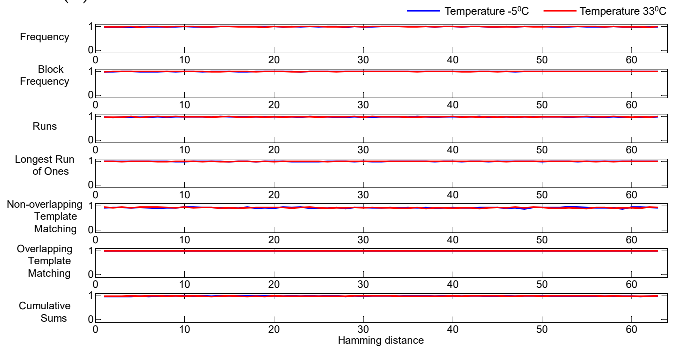
As mentioned in Section 4.3.6, we include both multiple interrogation (ST2) and parametric interrogation (ST3) schemes in the CRP generation process to enhance stability of the PUF responses. We also include ST2 in the BIST scheme to assess the improvements provided by these two tests. We assess the stability of PUF in the following three cases:



(a) Result of UT1: Randomness of Response



(b) Result of UT2: Effect of Individual Bit Transition



(c) Result of UT2: Effect of Multiple Bit Transitions

Fig. 8: Evaluation of Unpredictability of PUF at $-5°$C and $33°$C

i CRP is generated without any validation: The challenge is applied only once and the response is recorded.
ii CRP is generated after validation by ST2: Challenge is applied several times and the response is generated by vote.
iii CRP is generated after validation by both ST2 and ST3: Challenge is first validated by active parametric interrogation then the response is generated by vote.

Figure 9 shows the results of ST2: Multiple Interrogation Test for the above three cases. As explained in Section 4.3.6, this test sums up the PUF responses for T_2 times. The stability in the plot is calculated as $2.max(S_R, T_2 - S_R)/T_2$. For S_R close to either T_2 or 0 stability is close to 1. For S_R close to $T_2/2$ stability is close to 0. We see from the plot that applying ST2 in case ii greatly enhance the stability of the PUF as compared to case i. However, the most stable responses are generated in case iii where both ST2 and ST3 are applied.

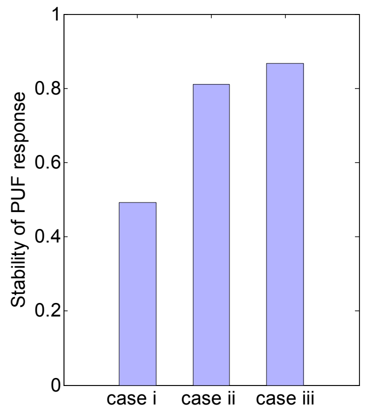


Fig. 9: Evaluation of PUF stability for three different CRP generation methods

6 Related Work

There have been a few studies on online hardware evaluation of true random number generators (TRNGs). Four tests (frequency test, poker test, run test, long run test) from FIPS 140-2 [36] were implemented on FPGA for hardware-based online testing of randomness in [11]. They evaluate two TRNG designs presented in [44] and [45] based on those four tests. In [12], 8 tests out of 15 NIST randomness tests were simplified for constant test sequence length and implemented in hardware (FPGA). [13] implement 9 tests from the NIST suite for on-the-fly testing of true random number generators. While operations like counting ones and zeros, finding the maximal longest run of the same value and keeping track of a random walk were implemented in hardware, all other operations including addition, multiplication, etc. were performed in a on-chip processor. The latter two work focus only on the implementation of the tests, they do not provide any TRNG evaluation results.

All the work described above is on testing the output bit sequence of ideal TRNGs. Unlike our BIST scheme, they do not test its internal health. A method to asses the internal health of an oscillator based TRNG by measuring the jitter was proposed in [17]. The work in [18] present an online scheme for evaluating the internal health of non-ideal TRNGs with non-IID internal sequences. Our BIST scheme tests both the output bits sequence and internal health and is generally applicable to any ideal TRNG design.

The work on the testing of security and stability of PUF is rather limited. The seminal work by Majzoobi et al. introduced the first formal methodology to test the security of strong PUFs in [7]. Four different tests are proposed: (i) predictability, (ii) collision, (iii) sensitivity, and (iv) reverse-engineering. Predictability test identifies the difficulty of correctly calculating or predicting the PUF output for a given input. Collision assessment studies how often two PUFs produce same outputs for an incident challenge. Sensitivity test ensures that the amount of process variation is sufficient for a PUF to remain stable when the operational, structural, and environmental conditions change. Reverse-engineering test determines the hardness of characterizing the PUF circuit component. These offline, software-based tests were important because they paved the way for understanding the PUF attack surface. They also enabled suggestions for novel input and output transformations for safeguarding the PUFs against attacks [4], [46].

Our BIST scheme adapts the predictability tests from [7] for the tests described in Section 3.2.1. However, the predictability tests were based on examining the probability of the output bit values and transitions, which is equivalent to the frequency test from NIST. We employ six more tests from NIST battery for a more comprehensive evaluation of unpredictability. The BIST scheme also includes the sensitivity tests for ST2 and ST3 described in Section 3.2.2. Beside a way to measure the stability, they are also made a part of the CRP generation process by online implementation. It greatly enhanced the stability of the PUF response.

The work by Hori et al. [9], and Maiti et al. [10] presented testing methodologies for quantitative analysis of different PUF designs. Maiti et al., besides defining their own parameters, analyzed the parameters proposed by several other work including [9] to determine any redundancy and defined a compact set of seven parameters: uniformity, reliability, steadiness, uniqueness, diffuseness, bit-aliasing, and probability of misidentification.

The measurement of uniformity is equivalent to the frequency test which is one of the seven standard randomness test included in our BIST scheme. Steadiness, a measure of stability, is similar to what we measure in the second stability test, ST2 described in Section 3.2.2. In both cases, a challenge is applied multiple times and the bias of the response bit towards either ’0’ or ’1’ is measured (more biased means more stable, irrespective of the direction). To measure reliability the responses to the same challenge is compared to the reference response. This requires saving the reference response on the chip which would create security threat. Therefore, this parameter is not suitable for on-line test.

The collision and reverse-engineering test from [7] and the last four parameters from [10] are different ways to measure the similarity (or dissimilarity) of two or more instances of PUFs. While these parameters are important for evaluating a PUF design, the BIST scheme proposed in this work is designed for real time monitoring of a single PUF instance, the design of which has already passed the initial evaluation. So these parameters are not relevant to our current work.

Evaluation of the weak PUF has been the subject of a number of earlier publications. Armknecht et al. provided theoretical analysis on robustness, physical unclonability and unpredictability properties of weak PUFs in [8]. Leest et al. used software-based Hamming weight test, interclass uniqueness test, context tree weighting test and NIST randomness tests to evaluate the randomness and entropy [47]. Cortez et al. tried to increase the fault coverage of physical faults, like stuck-at-fault, of Fuzzy Extractor (FE), which is the main component of weak PUFs, and proposed a secure BIST scheme to perform the fault tests [48].

To the best of our knowledge, BIST-PUF [21] is the first online and hardware-based testing methodology that includes higher order NIST randomness tests (other than the frequency test) and is applicable to all PUF families (weak and strong).

7 Conclusion and Future Directions

This paper presents a BIST scheme for online evaluation of PUFs and TRNGs. This is the first online hardware evaluation scheme for PUFs. The rising importance of PUF and its applications in several security and cryptography methodologies, highlight the need for analysis and evaluation of the PUF input-output behavior. The IoT platform especially needs PUF with self test capability to ensure seamless and secure operation as a large number of devices operate autonomously in a complex framework. The BIST structure is designed to evaluate two main properties of PUFs, namely *unpredictability* and *stability*. The unpredictability tests investigate the randomness of the response bits and the correlation between the transitions of response bit and challenge bits. The stability tests study the CRP behavior and the chip conditions in three distinct ways namely, sensor-based, parametric interrogation, and multiple interrogations.

While a number of work have dealt with online evaluation of TRNGs all of them were concerned with only testing statistical quality of the TRNG output. Even though these tests are important, monitoring the internal health of the TRNG is more crucial to ensure its integrity. Our BIST scheme is the first online test that, besides evaluating the output bit sequence, continuously monitors the internal health of ideal TRNGs and application of which is not limited to any specific TRNG design.

Other than the testing components, the BIST architecture also includes a low-overhead control circuitry and memory for saving the test results. Proof-of-concept implementation on FPGA demonstrated an overhead which is reasonable in terms of the percentage of the total used area of FPGA. We also provide evaluation results for a number of TRNG and PUF designs.

Going forward, there is a lot of room for continuing along the lines of research suggested in this work. A natural extension is working on built-in-self-repair (BISR) mechanisms that use the test results from the BIST scheme for on-the-fly improvement of randomness (entropy) and robustness (stability). A simple instance of such an improvement is provided by the stability tests: multiple interrogation and active parametric interrogation. The metastability based TRNG also employs a feedback mechanism which is essentially the frequency test from NIST or Diehard. Development of more complex repair methodologies is desirable.

Another important direction is development of new tests that can evaluate the susceptibility of TRNGs and PUFs to specific attacks. Devising attacks and countermeasures for modeling/reverse-engineering of PUFs is an active area of research [5], [6], [26], [32], [33]. Yet another possibility is to devise more efficient and compact tests for evaluations of PUF stability and robustness. While this paper focuses on strong PUF testing that is also applicable to several weak PUFs, it is interesting to extend the devised methodologies by applying them to other families of physical unclonable functions such as public PUF [49], processor-based strong PUFs that use aging for response tuning [50], or the FPGA-based time-bounded PUF that utilize the concept of erasability [51].

Acknowledgments

This work was supported in parts by an Office of Naval Research grant (ONR-R17460), National Science Foundation grants (CNS-1059416) and (CCF-1116858) and Air Force Office of Scientific Research grant under Multidisciplinary Research Program of the University Research Initiative (FA9550-14-1-0351).

References

[1] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of things: Vision, applications and research challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, 2012.

[2] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of iot systems: Design challenges and opportunities," in *IEEE/ACM International Conference on Computer-Aided Design*, ser. IEEE/ACM International Conference on Computer-Aided Design.

[3] M. Rostami, F. Koushanfar, and R. Karri, "A primer on hardware security: Models, methods, and metrics," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1283–1295, 2014.

[4] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure pufs," in *IEEE/ACM International Conference on Computer-Aided Design*, 2008, pp. 670–673.

[5] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *CCS*. ACM, 2010, pp. 237–249.

[6] U. Rührmair, X. Xu, J. Sölter, A. Mahmoud, M. Majzoobi, F. Koushanfar, and W. Burleson, "Efficient Power and Timing Side Channels for Physical Unclonable Functions," in *CHES*. Springer, 2014, pp. 476–492.

[7] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Testing techniques for hardware security," in *ITC*, 2008, pp. 1–10.

[8] F. Armknecht, R. Maes, A. Sadeghi, O.-X. Standaert, and C. Wachsmann, "A formalization of the security features of physical functions," in *IEEE S&P (Oakland)*, 2011, pp. 397–412.

[9] Y. Hori, T. Yoshida, T. Katashita, and A. Satoh, "Quantitative and statistical performance evaluation of arbiter physical unclonable functions on FPGAs," in *ReConFig*. IEEE, 2010, pp. 298–303.

[10] A. Maiti, V. Gunreddy, and P. Schaumont, "A systematic method to evaluate and compare the performance of physical unclonable functions," in *Embedded Systems Design with FPGAs*. Springer, 2013, pp. 245–267.

[11] R. Santoro, O. Sentieys, and S. Roy, "On-line monitoring of random number generators for embedded security," in *ISCAS*, 2009, pp. 3050–3053.

[12] F. Veljkovic, V. Rozic, and I. Verbauwhede, "Low-cost implementations of on-the-fly tests for random number generators," in *Design, Automation Test in Europe Conference Exhibition*, 2012, pp. 959–964.

[13] B. Yang, V. Rožić, N. Mentens, W. Dehaene, and I. Verbauwhede, "Embedded hw/sw platform for on-the-fly testing of true random number generators," in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*. EDA Consortium, 2015, pp. 345–350.

[14] W. Schindler and W. Killmann, "Evaluation criteria for true (physical) random number generators used in cryptographic applications," in *CHES*. Springer, 2012, pp. 431–449.

[15] W. Killmann and W. Schindler, "A proposal for: Functionality classes for random number generators." AIS, 2011.

[16] E. Barker and J. Kelsey, "Recommendation for the entropy sources used for random bit generation," *Draft NIST Special Publication*, 2012.

[17] V. Fischer and D. Lubicz, "Embedded evaluation of randomness in oscillator based elementary trng," in *Cryptographic Hardware and Embedded Systems*. Springer, 2014, pp. 527–543.

[18] B. Yang, V. Rozic, N. Mentens, and I. Verbauwhede, "On-the-fly tests for non-ideal true random number generators," in *IEEE International Symposium on Circuits and Systems*, 2015.

[19] M. Majzoobi, F. Koushanfar, and S. Devadas, "FPGA PUF using programmable delay lines," in *IWIFS*, 2010, pp. 1–6.

[20] K. Wold and C. H. Tan, "Analysis and enhancement of random number generator in fpga based on oscillator rings," *International Journal of Reconfigurable Computing*, vol. 2009, p. 4, 2009.

[21] S. U. Hussain, S. Yellapantula, M. Majzoobi, and F. Koushanfar, "Bist-puf: online, hardware-based evaluation of physically unclonable circuit identifiers," in *2014 IEEE/ACM International Conference on Computer-Aided Design*. IEEE.

[22] B. Sunar, W. J. Martin, and D. R. Stinson, "A provably secure true random number generator with built-in tolerance to active attacks," *Computers, IEEE Transactions on*, 2007.

[23] A. Cherkaoui, V. Fischer, A. Aubert, and L. Fesquet, "A self-timed ring based true random number generator," in *Asynchronous Circuits and Systems (ASYNC), IEEE 19th International Symposium on*. IEEE, 2013.

[24] H. Hisashi and S. Ichikawa, "Fpga implementation of metastability-based true random number generator," *IEICE TRANSACTIONS on Information and Systems*, vol. 95, no. 2, pp. 426–436, 2012.

[25] M. Majzoobi, F. Koushanfar, and S. Devadas, "FPGA-based true random number generation using circuit metastability with adaptive feedback control," in *CHES*. Springer, 2011.

[26] U. Ruhrmair, S. Devadas, and F. Koushanfar, "Security based on physical unclonability and disorder," M. Tehranipoor and C. Wang, Eds. Springer, 2011.

[27] J. W. Lee, D. Lim, B. Gassend, G. E. Suh, M. Van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *Symp. on VLSI*. IEEE, 2004, pp. 176–179.

[28] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *CCS*, 2002, pp. 148–160.

[29] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Design Automation Conference*. ACM, 2007, pp. 9–14.

[30] G. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Design Automation Conference*, 2007, pp. 9–14.

[31] M. Majzoobi, A. Kharaya, F. Koushanfar, and S. Devadas, "Automated Design, Implementation, and Evaluation of Arbiter-based PUF on FPGA using Programmable Delay Lines," Cryptology ePrint Archive, Report 2014/710, 2014, http://eprint.iacr.org/.

[32] A. Mahmoud, U. Rührmair, M. Majzoobi, and F. Koushanfar, "Combined Modeling and Side Channel Attacks on Strong PUFs," Cryptology ePrint Archive, Report 2013/632, 2013, http://eprint.iacr.org/.

[33] M. Rostami, J. B. Wendt, M. Potkonjak, and F. Koushanfar, "Quo vadis, PUF? trends and challenges of emerging physical-disorder based security," in *Design, Automation Test in Europe Conference Exhibition*, 2014, p. 352.

[34] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," NIST, Tech. Rep. 800-22, 2001.

[35] G. Marsaglia. (1995) Diehard battery of tests of randomness. [Online]. Available: http://www.stat.fsu.edu/pub/diehard/

[36] "FIPS 140-2: Security requirements for cryptographic modules," *Information Technology Laboratory, National Institute of Standards and Technology*, 2001.

[37] E. B. Barker and J. M. Kelsey, *Recommendation for Random Bit Generator (RBG) Constructions*. US Department of Commerce, NIST, 2012.

[38] ——, *Recommendation for random number generation using deterministic random bit generators (revised)*. US Department of Commerce, NIST, 2007.

[39] W. Schindler, "Functionality classes and evaluation methodology for deterministic random number generators," *Anwendungshinweise and Interpretation (AIS)*, pp. 5–11, 1999.

[40] "Wald-wolfowitz runs test," http://www.itl.nist.gov/div898/handbook/eda/section3/eda35d.htm, 2012.

[41] R. B. Davies, "Exclusive or (xor) and hardware random number generators," 2002. [Online]. Available: http://www.robertnz.net/pdf/xor2.pdf

[42] K. M. Zick and J. P. Hayes, "On-line sensing for healthier fpga systems," in *FPGA*, 2010, pp. 239–248.

[43] "Xpower analyzer," http://www.xilinx.com/support/documentation/sw_manuals/xilinx11/ug733.pdf, 2010.

[44] D. Schellekens, B. Preneel, and I. Verbauwhede, "Fpga vendor agnostic true random number generator," in *Field Programmable Logic and Applications, 2006. FPL '06. International Conference on*.

[45] V. Fischer and M. Drutarovskỳ, "True random number generator embedded in reconfigurable hardware," in *Cryptographic Hardware and Embedded Systems-CHES 2002*. Springer.

[46] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Techniques for design and implementation of secure reconfigurable PUFs," *ACM Trans. Reconfigurable Technologies and Systems (TRETS)*, vol. 2, no. 1, pp. 5:1–5:33, 2009.
[47] V. van der Leest, G.-J. Schrijen, H. Handschuh, and P. Tuyls, "Hardware intrinsic security from D flip-flops," in *STC*. ACM, 2010, pp. 53–62.
[48] M. Cortez, G. Roelofs, S. Hamdioui, and G. Di Natale, "Testing puf-based secure key storage circuits," in *Design, Automation Test in Europe Conference Exhibition*, 2014, p. 194.
[49] M. Potkonjak and V. Goudar, "Public physical unclonable functions," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1142–1156, Aug 2014.
[50] J. Kong and F. Koushanfar, "Processor-based strong physical unclonable functions with aging-based response tuning," *IEEE Trans. on Emerging Topics in Computing,*, vol. 2, no. 1, pp. 16–29, March 2014.
[51] M. Majzoobi and F. Koushanfar, "Time-Bounded Authentication of FPGAs," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 6, no. 3-2, pp. 1123–1135, 2011.

Siam U. Hussain Siam U. Hussain received his BSc in Electrical and Electronic Engineering from Bangladesh University of Engineering & Technology in 2011. He received his MSc in Electrical and Computer Engineering from Rice University in 2015. He worked at Samsung Bangladesh R&D Center as a software engineer from 2011 to 2013. He is currently a doctoral student at the Department of Electrical and Computer Engineering at University of California, San Diego. His research interests include hardware security, and privacy preserving computing.

Mehrdad Majzoobi Mehrdad Majzoobi received his MSc and PhD in Electrical and Computer Engineering from Rice University in 2009 and 2012 respectively. He is currently the chief executive officer of Mesh Motion Inc., a San Francisco based company that provides transportation technology as services for shared use mobility applications.

Farinaz Koushanfar Farinaz Koushanfar (M'03, SM'14) is a Professor and Henry Booker Faculty Scholar of Electrical and Computer Engineering (ECE) at the University of California, San Diego, CA, USA. From 2006-2015, She was an Assistant, Associate, and Full Professor of ECE in Rice University, TX, USA, where she also directed the Texas Instruments (TI) DSP Leadership University Program. She received her PhD in Electrical and Computer Engineering, and MA in Statistics in 2005, both from the University of California, Berkeley, CA, USA.

Her research interests include adaptive and low power embedded systems design, hardware security, and design intellectual property protection. Dr. Koushanfar is a recipient of several awards and honors, including the Presidential Early Career Award for Scientists and Engineers, the ACM SIGDA Outstanding New Faculty Award, the NAS Kavli Foundation Fellowship, and the Young Faculty (or CAREER) Awards from the Army Research Office (ARO), Office of Naval Research (ONR), Defense Advanced Research Projects Agency (DARPA), and National Science Foundation (NSF).