BIST-PUF: Online, Hardware-based Evaluation of Physically Unclonable Circuit Identifiers

Siam U. Hussain
and Sudha Yellapantula
ECE Department, Rice University
Email: {siam.umar,sudha}@rice.edu

Mehrdad Majzoobi
Mesh Motion Inc.
Berkeley, CA
Email: m.majzoobi@gmail.com

Farinaz Koushanfar
ECE Department
Rice University
Email: farinaz@rice.edu

Abstract– Physical Unclonable Functions (PUF) are of increasing importance due to their many hardware security applications including chip fingerprinting, metering, authentication, anti-counterfeiting, and supply-chain tracing, e.g., DARPA SHIELD. This paper presents BIST-PUF, the first built-in-self-test (BIST) methodology for online evaluation of weak and strong PUFs. BIST-PUF provides a paradigm shift in the evaluation of the unclonable circuit identifiers: unlike earlier known PUF evaluation suites that are software-based and offline, BIST-PUF enables on-the-fly assessment of the desired PUF properties all in hardware. More specifically, the BIST-PUF structure is designed to evaluate two main properties of PUFs, namely *unpredictability* and *stability*. These properties are important for ensuring robustness and security in face of operational, structural, and environmental fluctuations due to variations, aging or adversarial acts. For BIST-PUF unpredictability evaluation, we identify and adopt the tests of randomness that are amenable to hardware implementation. For stability assessment, the BIST-PUF suggests three distinct methods, namely, sensor-based, parametric interrogation, and multiple interrogations. Proof-of-concept implementation of the BIST-PUF in FPGA demonstrates its low overhead, effectiveness, and practicality.

Keywords: ***Built In Self Test, Physical Unclonable Functions, PUF, BIST, Hardware security, DARPA SHIELD***

I. Introduction

Identifying and tracing of individual fabricated silicon chips throughout their lifetime is a challenge. To keep a low cost and complexity of manufacturing, a single mask (blueprint) is used for mass production of integrated circuits (ICs) which cannot be uniquely identified [1], [2], [3], [4]. Classic IC identification and tracing 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 of the chips is important in that it can enable a low overhead identification, fingerprinting, authentication, metering, and tracing of semiconductor components along the untrusted supply chain [5].

The key idea behind 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 (partially) disordered physical system: when interrogated by a challenge (or input, stimulus), it generates a unique device response (or output). The response shall depend on the incident challenge, specific physical disorder and PUF device structure. It is common to call an input and its corresponding output a *challenge-response pair (CRP)*. Weak PUFs have a limited number of CRPs, while strong PUFs are capable of generating an exponential number of CRPs [6].

A recent timely solicitation by DARPA has called for innovative research proposals for an IC SHIELD that enables advanced supply chain hardware authentication capability [7]. While PUFs are promising enablers for generation of inherent and indelible chip IDs in several hardware security solutions including SHIELD, they cannot guarantee a fully stand-alone solution as the root-of-trust for ICs without a careful analysis. As shown several times in practice, when 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 [8], [9], [10].

To date, there have been very few works on the evaluation of usability (stability or repeatability) and security (randomness) properties of PUFs. A comprehensive methodology to test the security of PUFs was first introduced by Majzoobi et al. in [11]. 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, NIST [12] or Diehard [13] tests. Later research efforts have defined a more formal set of properties to evaluate PUFs, e.g., [14]. However, 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 BIST-PUF, the first methodology for online hardware-based assessment of the robust generation of streams of truly random (unpredictable) CRPs that are unique to each PUF device. Two main characteristics of PUF are evaluated by the BIST-PUF: 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. The continuous and online monitoring of PUF characteristics by the BIST-PUF yields several advantages including: (i) low overhead 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) reporting the exact conditions in the local PUF test site for a more granular debugging, and (iv) enabling active adjustment and improvements of the PUF operations.

Our main contributions are as follows.

- We design the first BIST for online assessment of the PUFs (in hardware) to quantitatively report and

978-1-4799-6278-5/14/$31.00 ©2014 IEEE

evaluate both the PUF stability and its security (unpredictability).

- For BIST-PUF unpredictability evaluation, we identify and adopt the tests of randomness that are amenable to hardware implementation. For BIST-PUF stability assessment we propose three distinct methods namely, sensor-based, parametric interrogation, and multiple interrogations.
- To remove the biases in challenge generation, our novel BIST-PUF architecture includes a high entropy TRNG module. Other than the testing components, the BIST-PUF architecture also includes a low-overhead control circuitry and memory for saving the test results.
- Proof-of-concept implementation of the BIST-PUF on FPGA demonstrates the effectiveness, practicability and low overhead of the proposed architecture.

Note that although BIST-PUF is applicable to both weak and strong PUFs, this paper focuses on strong PUF testing since it is a more general case. The rest of the paper is organized as follows. In the next section we outline the basics of PUFs and standard tests for randomness. The related literature is surveyed in Section III. In Section IV we discuss the novel BIST-PUF methodology and architecture. Section V describes the details of our implementation and the BIST-PUF evaluation results. The paper concludes in Section VI.

II. Background

A. Physical Unclonable Function

To make sure that a PUF is secure and stable it must posses the following properties [11]:

- *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 form 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 criteria 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 CRPs 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 [9]; (ii) one may use the side-channel information at various operational points to break the PUF security as demonstrated by our work [15], [10]. 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.

Note that although BIST-PUF 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. For more information about the PUF, its properties, and recent directions, we refer the interested readers to comprehensive articles on this topic [6], [16].

1) PUF implementation: A number of different realizations of the strong PUFs have been reported to date. In an *Arbiter PUF* [17], [18] 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 [19] the response from several Arbiter PUF stages were XORed to generate the PUF response. A *Lightweight Secure PUF* [20] has a similar structure, but the input challenge is passed through a complicated mapping to increase security against modeling attacks. We evaluate an improved version of the lightweight secure PUF presented in [20] with our BIST scheme. To implement the FPGA PUF we use the comprehensive methodology introduced in [21], [22].

B. Standard Randomness Tests

Batteries of randomness tests were originally designed to evaluate the performance of the random number generators. As discussed above, randomness of the response bits of PUF are mandatory to ensure that the PUF cannot be modeled as a deterministic process. Here we utilize the standard randomness test suits to evaluate the unpredictability of PUF response. Several standard test suites such as NIST [12], DIEHARD [13] , AIS.31 [23] are available for this purpose. A number of FPGA implementations of the randomness tests are reported in the literature. Four relatively simpler test from AIS.31 [24] are implemented in [25]. In [26] four DIEHARD tests whose implementation requires similar structures are selected so that execution time is reduced. In [27] , eight simple tests from NIST are chosen and made even simpler by mathematical manipulation to achieve both fast operation and low area. BIST-PUF adopts this latter design.

III. Related Work

The work on the testing of security 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 [11]. 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 such that a PUF is stable when the operational, structural, and environmental conditions change. Reverse-engineering 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 of novel input and output transformation for safeguarding the PUFs against attacks [20], [28].

Some more extensions and implementation of the testing methodologies presented in [11] were suggested in later works. In particular, Maiti et al. extended the work in PUF evaluation by some additional parameters: reliability, bit-aliasing, and probability of misidentification [29]. They also analyzed the parameters proposed by several other authors to determine any redundancy and tried to define a compact set. Note that the unpredictability assessment methods in [11] and its extensions were based on examining the probability of the output bit values and transitions, which is equivalent to the frequency test form NIST or Diehard tests. None of these earlier, software-based evaluations apply other standard randomness tests to PUF.

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 [14]. Leest et al. used software-based Hamming weight test, inter-class uniqueness test, context tree weighting test and NIST randomness tests to evaluate the randomness and entropy [30]. 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 [31].

To the best of our knowledge, BIST-PUF 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).

IV. Architecture of the BIST Scheme

The BIST scheme presented in this paper evaluates the PUF under test based on the two major properties described earlier: unpredictability and stability. The overall architecture of the proposed BIST-PUF is shown in Fig. 1. The finite state machine (FSM) controls the aforementioned two test sets, each of which consisting of three tests that shall be described in Sections IV-A and IV-B. The BIST generates the challenges as appropriate for each test. In addition to outputing N-bit challenges to the PUF input, the BIST output also includes the appropriate K-bit PUF tuning parameter. The input to the BIST block is the PUF output(s) or responses. The tests themselves are performed by the block denoted as "hardware randomness tests" in the figure. There is also a random challenge generation component. The exact implementation details of the figure blocks is outlines in Section V where we describe the hardware overhead and implementation of the BIST-PUF.

A. Unpredictability Test

The unpredictability tests checks the randomness of the response from the PUF. We exploit the standard tests from the NIST battery of randomness test to evaluate the randomness. The test module resides on the same chip as the PUF. Each test operates on a block of response bits and outputs one for success and zero for failure. Unpredictability is evaluated in the three following ways, denoted by UT 1.1, UT 1.2, and UT 1.3 respectively.

UT 1.1: 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.

UT 1.2: 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 N-bit random challenge vector. The PUF is then interrogated by a similar vector where only the i-th bit is inverted $i \in \{1, 2, \ldots, N\}$. 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 ($i = 1, 2, \ldots, N$) and the success rate is reported as a function of the inverted bit index, i.

UT 1.3: 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 N-bit random challenge vector. The PUF is then evaluated with a similar vector where h bits are inverted. Thus, the Hamming distance between these two challenge vectors is h, where $h \in \{1, 2, \ldots, N\}$. 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.

B. 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. Basically, an evaluation methodology would sense the condition, learn the effect of the condition on the PUF CRPs through preliminary analysis, and adjust the responses for the particular mode. Hardware-based sensors are very useful for comprehending the mode of operation and they form our first stability testing method.

The second proposed methodology for stability testing is based on changing the parameter(s) affecting the stability and then check the impact on the response. We demonstrate a recent representative evaluation method for this category. The third and last stability testing method 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.

1) 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

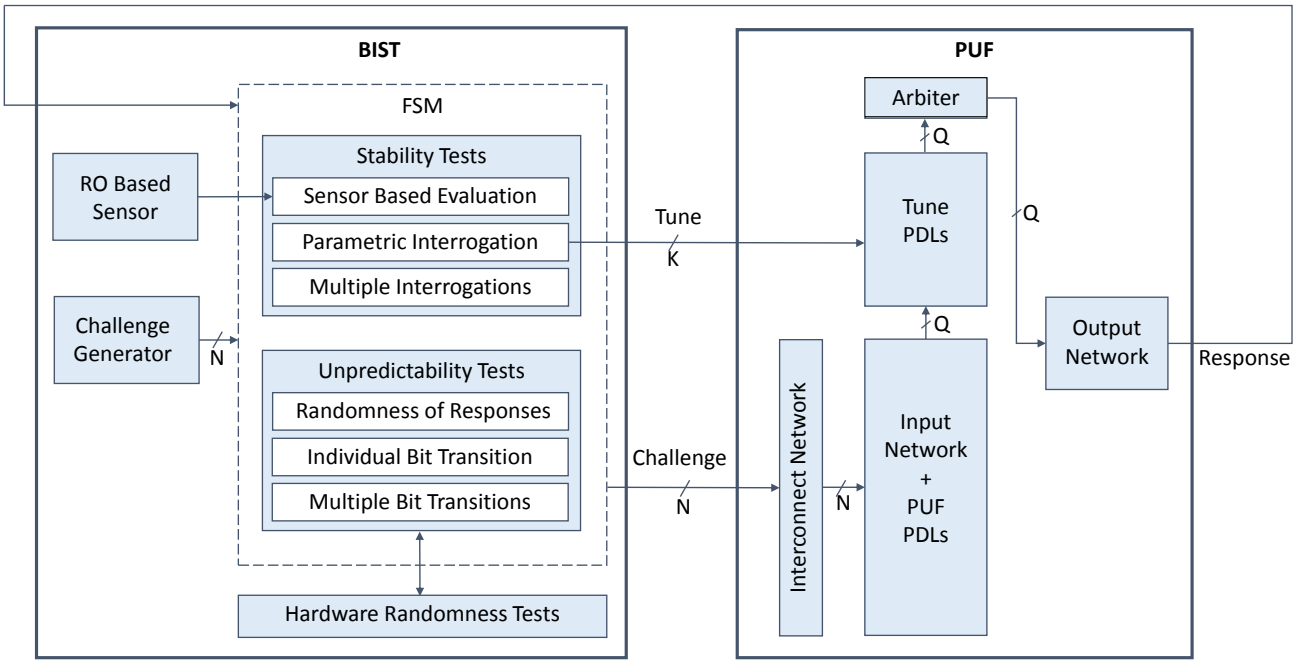

Fig. 1: Architecture of the BIST-PUF Scheme

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.

Note that the sensors that can detect mechanical invasive attacks or optical exploits are also of increasing significance in tamper detection, for e.g. SHIELD [32], [33]. A discussion of these sensors is outside the scope of the present paper.

Many modern ICs already include a (built-in) temperature sensor. However, one temperature sensor can only report the average chip temperature and is unable to report the granular temperature fluctuations throughout the chip. Although there are a number of proposals for on-chip voltage sensors in the literature, but they are more complicated to implement and less used.

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 a RO.

2) Active Parametric Interrogation: The PUF randomness is typically converted to a digital binary format using a metastable component, for e.g., arbiter metastability. A random parameter or a set of parameters drive the point of operation of this metastable part. For instance, in delay based PUF, the random parameter(s) are the delays of the PUF components that are combined using the challenge set. Changing this random parameter by a small amount can potentially alter the operating mode of the metastable component and yield important information/confidence about the robustness of operation. In parametric interrogation, this random parameter is altered in deterministic steps around its expected value and the resulting response is observed.

An example of parametric interrogation for arbiter-based PUF was suggested and evaluated in [22]. For the sake of completeness, we briefly describe this procedure. The underlying hypothesis is that a larger delay difference at the arbiter input leads to a more robust (stable) response. To parametrically change the delay at the arbiter input, a programmable delay line is used which changes the delay difference between the higher and lower arbiter input in multiple steps of size $\pm\Delta_t$. If the response of the arbiter after several steps stays the same, then the response is stable with a high-confidence and the incident challenge is marked as a *robust challenge*. If the response of the arbiter is sensitive and unstable with respect to the parametric changes to the arbiter input, then the response confidence and stability are lower. This type of parametric delay testing is interesting since given a good step size and number of steps, it can yield a quantitative measure of stability for each response. In [22] this methodology was also used to a classify and group challenges into different robustness sets.

3) 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 [22] 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.The reduction in the metastable window width is logarithmic with respect to the number of repetitions.

V. Implementation and Evaluation

We implement the proposed BIST-PUF on a Xilinx Virtex 6 FPGA. The PUF under test is a delay-based strong PUF proposed in [21]. It also incorporates the input and interconnect networks introduced by the lightweight secure PUF in [8]. The output network is just an XOR mapping with Q' bits of input and a 1 bit output. The BIST component resides on the same chip as the PUF and is able to run its tests automatically. An overview of the BIST-PUF implementation is displayed in Fig. 1 in Section IV.

A. PUF

The principles of the delay based PUF were first introduced by Gassend et al. in [18]. In this PUF, generating one bit of output 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 at the end of the two parallel path that generates the response accordingly. The PUF challenges (inputs) act as the selector bits of the switches.

In our FPGA implementation, the switches are realized by the programmable delay lines (PDL) as described in [21], [22]. A PDL is an LUT 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. The path with a larger delay is equivalent to the cross connector in the switch. PDLs are also used to compensate for the bias in a path delay caused by the asymmetry in signal routing.

The complete PUF circuit with the switch structure and the tuning blocks is shown in Fig. 2. The triangular elements represent PDLs. The shown system consists of N switches and K tuning blocks. In our implementation, we use $N = 64$ and $K = 16$. The tuning blocks insert 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 PDLs in a switch block. To increase the unpredictability (randomness) in the PUF response and make it more secure, responses from $Q(= 16)$ parallel PUFs are combined.

B. Peripheral Circuitry

The peripheral circuitry adds security to the PUF by using input/output transformations that thwart active attacks. It consists of the following components:

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 one LUT. As explained in [34] XOR operation reduces the bias present in a set of random sequences if they are independent and uncorrelated.

3) Interconnect Network: The interconnect network connects the challenge bits to the rows of the parallel arbiter-based PUF lines. The interconnection rule can be expressed formally as follows.

$$c_i^m = c_j^{m+1} \ for\, j \in \Omega \ and\ m = 1, 2, \ldots, (Q-1), \tag{4}$$

where, c_i^m is the i-th challenge bit in the m-th row, $\Omega = 1, 2, \ldots, N$, and $j = g_m(i)$, $g : \Omega \rightarrow \Omega$ is a one-to-one permutation function. By imposing a constraint on g_m to be non-identity for all m's, it can be ensured that it is not possible to fully bypass more than one input network. We set g_m to be

$$j = g_m(i) = (i + m - 1) \mod Q. \tag{5}$$

C. BIST Architecture

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

1) Challenge Generator: To perform one round of the randomness tests described in Section IV, 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 [35] which is an enhancement of the Sunar type random number generator [36]. Our TRNG comprises of 32 ring oscillators each of which contains three inverters. 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.

2) Randomness Test Module: The randomness test module incorporates seven tests from the NIST test suit. 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 [27]. Equations (6) and (7) show the general form of

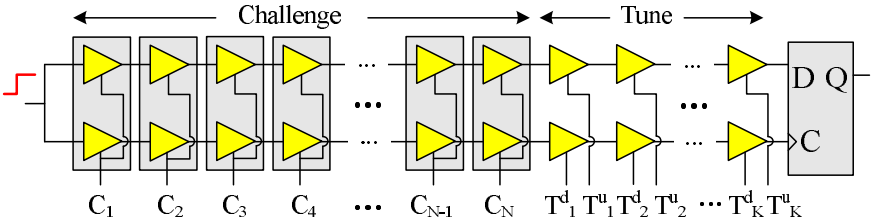

Fig. 2: Internal structure of the PDL based PUF.

simplifications on the *erfc* and *igamc* functions respectively.

$$P\_value = erfc(x)$$
$$P\_value > \alpha => x < erfcinv(\alpha) \quad (6)$$
$$P\_value = igamc(a, x)$$
$$P\_value > \alpha => x < igamcinv(a, (1-\alpha)) \quad (7)$$

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.

3) RO Based Sensor : Zick et al. presented a compact RO based sensor for online measurements of variations in delay [37]. 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) FSM: The FSM is designed to control the tests described in Sec IV. The unpredictability tests requires the randomness test module. To optimize the area, we designed the FSM such that it performs these three tests on the PUF sequentially using the same module. The random challenge vectors from the challenge generator is read by the FSM which performs the necessary manipulations on them as required by the tests UT 1.2 and UT 1.3. The response from the PUF goes to the test module via the FSM. 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 PUF performance using a more comprehensive data.

Our sensor based evaluation reads the delay measurement data from the RO-sensor. For active parametric interrogation test, the FSM changes the tuning parameters of the PUF for a fixed challenge while the results are accumulated. The responses are read with three values of the tuning parameter that (i) creates extra delay on top path, (ii) creates extra delay on bottom path, and (iii) keeps both delays similar. If the sum of responses is either 0 or 3, the challenge is noted as robust. The number of robust challenges and their values will be stored in the memory. For multiple interrogation tests, the tuning parameters are set to the optimum value. 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.

Component	**Register/ LUT/ Slice**	**Power(mw)**
PUF with Peripheral Circuitry	16/ 1558/ 768	18.92
Challenge Generator	144/ 492/ 184	3.49
Randomness Test Module		
1) Frequency	46/ 64/ 27	0
2) Block Frequency	32/ 50/ 20	1.88
3) Runs	64/ 138/ 51	0.04
4) Longest Run of Ones	149/ 307/ 119	0.4
5) N.O. Template Matching	718/ 836/ 296	1.55
6) Overlapping Template Matching	30/ 35/ 17	1.55
7) Cumulative Sums	77/ 136/ 59	0.45
RO-Sensor	8/ 14/ 4	0
FSM	112/ 418/ 128	25.97

TABLE I: Resources usage and reaction time of the NIST tests implemented on FPGA.

D. Overhead Estimation

The resource and power consumption by different parts of the BIST-PUF is reported in Table I. The power consumption by the FSM is about 7 mW larger than that of the PUF. But it should be noted that the BIST scheme need not always run in parallel to the PUF. For most applications it is sufficient to sporadically audit the PUF performance using the BIST module.

E. Randomness Test Results

For randomness testing, we implement the arbiter-based PUF on 12 Xilinx Virtex 5 (LX110) FPGAs. The exact details of this implementation can be found in [22]. Each FPGA includes 16 PUF rows with parallel racing paths. Each row of the PUF generates 64K responses to 64K unique random challenges generated by the TRNG. These PUF responses have been tested for randomness using the hardware-implemented NIST Test Suite (outlined in the first column of Table I). The randomness test results are reported in Table II. The columns contain the six randomness tests that we have applied. The test names are shown in the table rows and are self-explanatory, except for the N.O. Template Matching, which refers to the Non-Overlapping Template Matching test. Each column describes the test results for one FPGA. The value shown in each table cell demonstrates the percentage of the response bits that pass the pertinent randomness test. The results show that the responses from the PUF are highly random with a very high degree of confidence.

Note that we have performed a number of other randomness tests for quantifying the effects of a single-bit and multi-

Randomness Tests	FPGA 1	FPGA 2	FPGA 3	FPGA 4	FPGA 5	FPGA 6	FPGA 7	FPGA 8	FPGA 9	FPGA 10	FPGA 11	FPGA 12
1) Frequency	99.1	98.8	98.1	99.5	98.3	98.6	98.9	98.6	98.3	99.3	99.1	99.7
2) Block Frequency	99.7	99.4	98.9	99.5	99.5	100.0	99.7	99.4	99.7	99.3	99.7	99.5
3) Runs	97.7	97.2	96.6	96.9	97.5	96.6	97.0	97.3	96.7	97.5	97.2	98.1
4) Longest Run of Ones	98.8	99.4	99.2	99.0	99.4	99.0	99.6	98.2	98.4	99.4	99.4	99.2
5) N.O. Template Matching	99.8	99.8	100.0	99.6	100.0	99.8	100.0	99.8	99.8	100.0	99.8	99.4
6) Cumulative Sums	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

TABLE II: Percentage of success of each Randomness Test for PUF responses collected from 12 FPGAs.

bits transitions in the PUF challenges. The randomness test results are promising. Due to space constraints, a full report of all the randomness tests is outside the scope of the current paper.

VI. Conclusion and future directions

Indelible signatures of the ICs extracted by physical unclonable functions (PUF) can enable identification, fingerprinting, metering, authentication, and tracing of the components along the unascertained semiconductor supply chain. Due to its ability to individually mark each (mass produced) IC, a PUF may be the only plausible solution for implementing the root-of-thrust for an IC SHIELD described in a recent DARPA solicitation [32], [7]. A carefully designed and safeguarded PUF also provides a low-overhead alternative to secure key storage and can be the basis of several modern security and cryptography protocols, e.g., [38].

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 BIST-PUF approach presented in this paper provides the first built-in-self-test (BIST) methodology for online and hardware-based evaluation of PUF. The BIST-PUF structure is designed to evaluate two main properties of PUFs, namely *unpredictability* and *stability*. The predictability 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.

To remove the biases in challenge generation, our novel BIST-PUF architecture includes a high entropy TRNG module. Other than the testing components, the BIST-PUF architecture also includes a low-overhead control circuitry and memory for saving the test results. Proof-of-concept implementation of the BIST-PUF in FPGA demonstrated the overhead of the testing components, as well as randomness evaluation results for a large number of responses collected from 12 FPGA boards. While this paper reports a number of promising preliminary FPGA implementation results from application of the suggested BIST-PUF evaluations, several of our test results are not included due to the space constraints.

Stability testing and/or improvement do not remove the requirement for performing error correction. Several security and cryptography applications of PUF (e.g., secret key generation) require an error-free operation which is hard to guarantee for physical systems. However, improving the stability would help by lowering the size and other overheads of error correction. Newer robust PUF protocols which use alternative methodologies for error correction instead of traditional error correction codes (ECC) could also benefit from our suggested BIST-PUF. An interesting example of such protocols is provided by the SlenderPUF protocol, which utilizes string-matching to correct for the errors in the PUF operation [39], [40]. Another important advantage of SlenderPUF protocol is that it increases the PUF's resistance against remote machine learning attacks. There is a need for development of more such robust protocols which extend the operability range of the PUFs.

Going forward, there is a lot of room for continuing along the lines of research suggested by the BIST-PUF. A natural extension is working on built-in-self-repair (BISR) mechanisms that use the test results from the BIST-PUF for on-the-fly improvement of randomness (entropy) and robustness (stability) of the responses. A simple instance of such an improvement is provided by the multiple interrogation stability test, which uses the consensus among the responses to a repeated challenge to increase stability. Development of more complex repair methodologies is desirable.

Another important direction is development of new tests that can evaluate the susceptibility of the PUF to specific attacks. Devising attacks and countermeasures for modeling/reverse-engineering of PUFs is an active area of research [6], [16], [9], [15], [10]. One way to attack the PUF is to perform measurements at multiple operational and environmental conditions, e.g., various temperature and supply voltage values [41]. Sensing the changes in these parameters along with reporting of the stability/predictability of the resulting responses can help in thwarting these attacks. 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 [42], processor-based strong PUFs that use aging for response tuning [43], or the FPGA-based time-bounded PUF that utilize the concept of erasability [44].

VII. Acknowledgment

This work was supported in parts by an Office of Naval Research grant (ONR-R17460) and National Science Foundation grants (CNS-1059416) and (CCF-1116858).

REFERENCES

[1] F. Koushanfar, G. Qu, and M. Potkonjak, "Intellectual property metering," *Information Hiding Workshop*, pp. 81–95, 2001 2001.

[2] Y. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security," in *USENIX Security Symp.*, 2007, pp. 291–306.

[3] J. Roy, F. Koushanfar, and I. Markov, "EPIC: Ending Piracy of Integrated Circuits," *IEEE Computer*, vol. 43, no. 10, pp. 30–38, 2010.

[4] F. Koushanfar, "Provably Secure Active IC Metering Techniques for Piracy Avoidance and Digital Rights Management," *IEEE Trans. on Information Forensics and Security (TIFS)*, vol. 7, no. 1, pp. 51–63, 2012.

[5] M. Rostami, F. Koushanfar, and R. Karri, "A primer on hardware security: Threat models, metrics, and remedies," *Proceedings of the IEEE*, pp. 1–13, 2014, to appear.

[6] U. Ruhrmair, S. Devadas, and F. Koushanfar, "Security based on physical unclonability and disorder," *Book Chapter in Introduction to Hardware Security and Trust*, 2011.

[7] Defense Advanced Research Projects Agency (DARPA), Microsystems Technology Office/MTO Broad Agency Announcement, "Supply chain hardware integrity for electronics defense (SHIELD)," 2014.

[8] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure pufs," in *ICCAD*. IEEE Press, 2008, pp. 670–673.

[9] 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.

[10] 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*, 2014.

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

[12] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," DTIC Document, Tech. Rep., 2001.

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

[14] 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.

[15] 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/.

[16] M. Rostami, J. B. Wendt, M. Potkonjak, and F. Koushanfar, "Quo vadis, PUF? trends and challenges of emerging physical-disorder based security," in *DATE*, 2014, p. 352.

[17] 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.

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

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

[20] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure pufs," in *International Conference on Computer-Aided Design (ICCAD)*, 2008, pp. 670–673.

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

[22] 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/.

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

[24] "Fips 140-2: Security requirements for cryptographic modules." [Online]. Available: http://csrc.nist.gov/publications/fips/fips140-1

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

[26] A. Vaskova, C. López-Ongil, E. San Millán, A. Jiménez-Horas, and L. Entrena, "Accelerating secure circuit design with hardware implementation of diehard battery of tests of randomness," in *International On-Line Testing Symposium (IOLTS)*, 2011, pp. 179–181.

[27] F. Veljkovic, V. Rozic, and I. Verbauwhede, "Low-cost implementations of on-the-fly tests for random number generators," in *DATE*, 2012, pp. 959–964.

[28] 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.

[29] 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.

[30] 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.

[31] M. Cortez, G. Roelofs, S. Hamdioui, and G. Di Natale, "Testing puf-based secure key storage circuits," in *DATE*, 2014, p. 194.

[32] F. Koushanfar and R. Karri, "Can the SHIELD protect our integrated circuits?" in *International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2014.

[33] J. Rajendran, D. Shahrjerdi, S. Garg, F. Koushanfar, and R. Karri, "Shielding and securing integrated circuits with sensors," in *ICCAD*, 2014.

[34] R. B. Davies, "Exclusive or (xor) and hardware random number generators," *Tech. Rep*, 2002.

[35] 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, 2009.

[36] B. Sunar, W. J. Martin, and D. R. Stinson, "A provably secure true random number generator with built-in tolerance to active attacks," *IEEE Transactions on Computers*, vol. 56, no. 1, pp. 109–119, 2007.

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

[38] J. Kong, F. Koushanfar, P. K. Pendyala, A.-R. Sadeghi, and C. Wachsmann, "PUFatt: Embedded Platform Attestation Based on Novel Processor-Based PUFs," in *DAC*, 2014, pp. 109:1–109:6.

[39] M. Majzoobi, M. Rostami, F. Koushanfar, D. S. Wallach, and S. Devadas, "Slender PUF Protocol: A lightweight, robust, and secure authentication by substring matching," in *International Workshop on Trustworthy Embedded Devices*, 2012.

[40] M. Rostami, M. Majzoobi, F. Koushanfar, D. Wallach, and S. Devadas, "Robust and Reverse-Engineering Resilient PUF Authentication and Key-Exchange by Substring Matching," *IEEE Trans. on Emerging Topics in Computing*, 2014.

[41] S. Wei, J. B. Wendt, A. Nahapetian, and M. Potkonjak, "Reverse Engineering and Prevention Techniques for Physical Unclonable Functions Using Side Channels," in *DAC*, 2014, pp. 90:1–90:6.

[42] M. Potkonjak and V. Goudar, "Public physical unclonable functions," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1142–1156, Aug 2014.

[43] 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.

[44] 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.