CSST: Preventing Distribution of Unlicensed and Rejected ICs by Untrusted Foundry and Assembly

Md. Tauhidur Rahman, Domenic Forte, Quihang Shi, Gustavo K. Contreras, and Mohammad Tehranipoor
ECE Department, University of Connecticut
{tauhid, forte, qihang.shi, gustavo.contreras, tehrani}@engr.uconn.edu

Abstract— **The globalization of the semiconductor design and fabrication industry (also known as the horizontal business model) has led to many well-documented issues associated with untrusted foundries and assemblies, including IC overproduction, cloning, and the shipping of improperly or insufficiently tested chips. Besides the loss in profits to Intellectual Property (IP) owners, such chips entering the supply chain can have catastrophic consequences for critical applications. We propose a new Secure Split-Test (SST) scheme called the Connecticut SST (CSST) in which the IP owner takes full control over testing. In CSST, each chip and its scan chains are locked during testing, and only the IP owner can interpret the locked test results and unlock passing chips. The new SST can prevent overproduced, defective, and cloned chips from reaching the supply chain. The proposed method considerably simplifies the communication required between the foundry/assembly and the IP owner compared to the original version of the SST. The results demonstrate that our new technique is more secure than the original and has lower communication overheads.**

I. Introduction

The continued scaling and complexity of integrated circuits (ICs) has significantly inflated manufacturing costs in the semiconductor industry. For example, moving from the 32nm to the 28nm technology node increased manufacturing costs by 40% [9]. And, the development and recurring maintenance costs of new semiconductor fabrication are expected to exceed US $ 5.0 billion by 2015 [1], [2]. In order to cope, many semiconductor companies have closed their fabrication plants [1] and have migrated to a fabless business model. In this model, the IP owner is susceptible to such threats as IP theft, mask theft, IC piracy, cloning, counterfeiting, and the inserting of malicious circuits (hardware Trojans) [2].

The typical supply chain steps (design, fabrication, and so on) and vulnerabilities are shown in Figure 1 [2]. The fabless design house produces the design (IP) and sends it to the foundry for fabrication in the form of a GDSII file, test patterns, and corresponding responses. GDSII is a geometric representation of the 3D structures of an IC that guides the fabrication process [9], [13]. After fabrication, the foundry tests the die on each wafer by comparing the test patterns - or responses - with the correct ones. Those die that pass are packaged by assembly and are then retested. The assembly is responsible for shipping only good ICs into the market. There are different vulnerabilities associated with each step in the horizontal supply chain. A GDSII file contains the whole IP design [14], [15]. Untrusted entities in the fabrication plant can tamper with the GDSII. Untrusted fabrication plants can also overproduce (produce more ICs than they are contracted to produce) and then send out-of-spec and defective ICs into the market (through lax or nonexistent testing). Overproduction, out-of-spec/defective, or tampered attacks are possible at the assembly site, as well.

There has been extensive research to combat the forms of IC theft, cloning, and counterfeiting committed by untrusted foundries. Active metering, logic obfuscation, secure split-testing, source code encryption, and bitstream encryption for FPGA are the major existing solutions used to mitigate these attacks [3]–[10], [13]. Many of these schemes rely on the encryption of combinational logic and/or a finite state machine (FSM) block performed via locking mechanisms [3]–[10], [13]. In the locking mechanism, only a valid specific input vector (i.e, a unique key) can lead the IC to its correct functionality. To achieve this, extra logic blocks are inserted in the main design, which only become transparent with a valid key. For example, a group of extra finite states are added in order to lock the FSM, and only the valid input sequence can bring the modified FSM to the correct initial state for normal working mode. Active metering [3]–[6] allows the IP owner to lock and unlock each IC remotely. The locking mechanism is a function of the unique ID generated for each IC by a Physically Unclonable Function (PUF, [11], [12]). Only the IP owner knows the transition table and can unlock the IC from this ID. In EPIC [7], each IC is locked with randomly inserted XOR gates. The XOR gates will only become transparent with the application of a valid key (effectively unlocking the IC). In this technique, a set of public/private keys needs to be generated by the IP owner, foundry, and each IC. The primary objective of these approaches is to give the IP owner control over the exact number of ICs that can enter the market by obfuscating the correct behavior.

However, the flaw in these approaches [3]–[6], [9], [10] is that, in the typical supply chain, the chips *must be unlocked before they are tested by the foundry and assembly*. Hence, the IP owner cannot truly control how many ICs enter the supply chain because theres nothing to prevent foundries and assemblies from pretending their yield is low during the test process and then requesting more keys. Furthermore, theres no guarantee that untrusted fabs or assemblies will perform IC testing correctly - or even at all. Such defective parts may exhibit correct functionality for the most part and can, therefore, be very difficult to spot in the supply chain. However, if integrated into a critical system, there could be serious consequences. To summarize, shortcomings in the prior approaches can potentially allow foundries and assemblies to ship overproduced and out-of-spec or defective ICs to the market. We extend [20] for both assembly and foundry in this paper.

We then proposed the Secure Split-Test (SST) [13] to

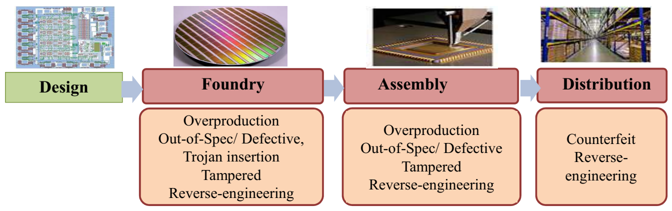


Fig. 1. Supply chain and vulnerability [2].

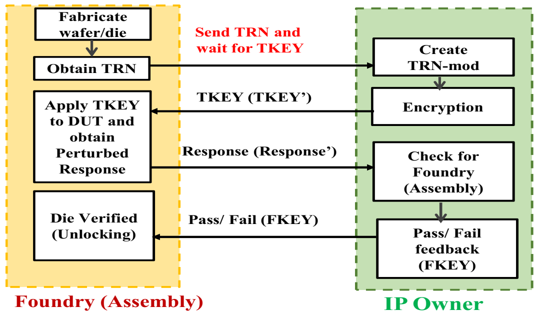


Fig. 2. Steps between foundry and assembly with IP owner in Secure Split-Test. Assembly steps that differ from foundry steps are contained in brackets. The wait for TKEY (TKEY') is a critical issue in SST.

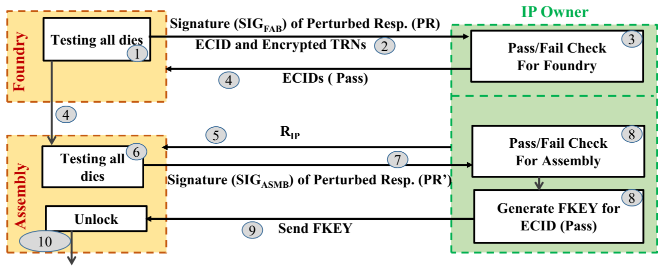

IP Owner
Foundry
Testing all dies
1
Signature (SIG_FAB) of Perturbed Resp. (PR)
ECID and Encrypted TRNs
2
Pass/Fail Check For Foundry
3
ECIDs (Pass)
4
4
Assembly
5
R_IP
Testing all dies
6
7
Signature (SIG_ASMB) of Perturbed Resp. (PR')
Pass/Fail Check For Assembly
8
Generate FKEY for ECID (Pass)
8
Unlock
9
Send FKEY
10

Fig. 3. CSST communication flow. The CSST removes the complex communication barrier with non-blocking communication between the foundry/assembly and the IP owner.

address these shortcomings and to help protect against the forms of IC piracy being performed by untrusted foundries and assemblies. The existing SST ensures that the IP owner has full control in the horizontal IC supply chain. But, the SST also involves large test times and complex flows of communication. In SST, foundries and assemblies need to communicate with the IP owner for each decision and in order to get keys for each die in the wafer. This blocking, multi-session communication between the IP owner and the foundry or assembly is a major obstacle because of the large timing overhead it requires. In this paper, we are proposing the Connecticut Secure Split-Test (CSST), which possesses the following advantages: 1) there is no communication barrier; 2) this method is more resistant to attacks (e.g., flush and shift attacks); and 3) there are tunable security-area trade-off parameters. The benefits of CSST can also be demonstrated by its $\sim 50\%$ hamming distance with tuning parameters and its resiliency against different potential attacks.

The rest of this paper is organized as follows: Section II summarizes the original SSTs work-flow. Section III describes the CSST technique and how its locking blocks are implemented. Section IV compares the SST and the CSST in terms of their relative security, area overhead, and vulnerability to attacks. Finally, we conclude the paper in Section V.

II. Background: The Secure Split-Test (SST)

The Secure Split-Test (SST) was designed to prevent IC overproduction, cloning, and the circulation of out-of-spec or defective counterfeit ICs in the electronics supply chain. In the SST, unlike the other techniques, only the IP owner determines whether an IC fails or passes testing. Figure 2 shows the original SSTs communication flow, where true random numbers (TRNs), test keys (TKEYs), responses, and functional keys (FKEYs) are exchanged between the IP owner and the foundry/assembly. In the SST, the foundry (assembly) has to communicate with the IP owner for each die. The foundry (assembly) obtains the TRN (TRN') from each die under test (DUT) and sends it to the IP owner and waits for a TKEY (TKEY') to perform testing on the DUT. The IP owner receives the TRN and modifies it in such a way that only the IP owner knows the modified TRN (TRN-mod). The TRN-mod (TRN-mod' for the assembly) is encrypted using RSA to generate TKEY (TKEY'). The IP owner sends the TKEY (TKEY') to the foundry (assembly). The foundry (assembly) receives the TKEY (TKEY') and applies it to the DUT. The DUT will decrypt the TKEY (TKEY') internally to obtain the TRN-mod (TRN-mod'). The functional-locking and scan-locking blocks use the TRN-mod to alter the IC functionality and the perturb test responses, respectively. A locked IC gives a different response than the original response and can only be unlocked by a valid FKEY sent to foundry or assembly. The functional-locking block prevents untrusted foundries or assemblies from accessing the ICs correct functionality. This portion of the SST is similar in spirit to prior work [3]–[6], [9], [10]. Besides the functional-locking block, the SST also introduces scan chains that are locked with a scan-locking block. The scan-locking block perturbs the test responses uniquely for each IC so that the foundry/assembly cannot guess the correct response from any unlocked IC. Because of the locking mechanism, only perturbed responses (responses') are obtained from the DUT by fabrication plants. The fab (assembly) sends them to the IP owner to check the result, and the IP owner decides whether that DUT passes/fails. If the IC passes, then the IP owner asks the foundry to package that IC at the assembly plant. And, if the IC passes in assembly, then the IP owner sends the FKEY to unlock that IC. This process is repeated for all chips. In the SST, instead of the foundry or assembly, it is the IP owner that checks the perturbed responses and decides whether an IC passes or fails. Then and only then does the IP owner send a key to unlock the IC. In this way, the SST prevents untrusted foundries and assemblies from sending defective, extra, or overproduced ICs into the market.

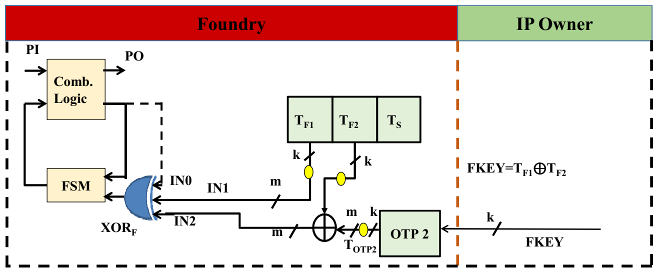
One of the SSTs most serious drawbacks is the wait for TKEY step, which must occur for each DUT at the foundry and assembly. This blocking communication step severely limits the SSTs practical acceptance in standard IC test flow. To address this, we are proposing a more efficient version of the Secure Split-Test called the Connecticut Secure Split-Test (CSST) which considerably reduces the test time and communication overhead involved in the SST. In addition, the CSST is even more secure than the original SST.

III. The Connecticut Secure Split-Test (CSST)

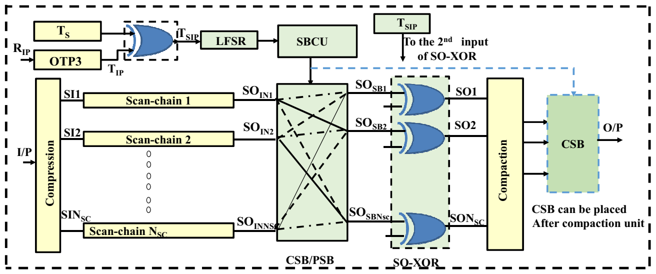
The CSSTs objective is the same as the SSTs. Detailed steps for the CSST are as follows (illustrated in Fig. 3):

1) At the foundry, each IC in a wafer generates a TRN and stores it internally in a one-time programmable (OTP) device. A functional-locking block uses the TRN to lock the functionality and a scan-locking block employs the TRN to internally perturb the scan chains outputs. The TRN value is unique for each IC, and this results in different perturbed responses for different ICs. Each IC encrypts its TRN value internally using the IP owner's public key and outputs the encrypted TRN value to the fab. Hence, the foundry does not know the unique TRN value, which prevents the foundry/assembly from guessing the true responses of the device.
2) The foundry applies test patterns to each wafer and collects the following information: the electronic chip IDs (ECIDs) of all die, the encrypted TRNs (ciphertext), and the signatures from the perturbed outputs of the scan chain. This information is sent to the IP owner.
3) With this information, the IP owner determines which die in the wafer pass or fail as follows: first, the TRN value is decrypted using the IP owner's private key. Since this key is only known to the IP owner, only the IP owner can determine the TRNs for each die. Next, the IP owner computes the signatures associated with these TRNs. Finally, the IP owner compares the signatures computed to those sent by the fab. Those die with signatures that match the IP owner's are considered fault-free.
4) The foundry marks the passing die based on the IP owner feedback and sends the wafers to assembly where the wafers are diced, packaged, and re-tested.
5) The IP owner sends a random number to the assembly that re-perturbs the test outputs. This step prevents the assembly from replaying the correct perturbed responses for passing ICs (obtained from the fab or before packaging) and sending them to the IP owner without testing the IC.
6) The assembly applies the random number to the IC and receives the corresponding response after testing. The response is different than the response from the foundry because of the IP owner generates random number.
7) The assembly sends the signature from the perturbed responses with the corresponding ECIDs to the IP owner for a decision. Data for all ICs are sent in a single session.
8) The IP owner checks the signature for each IC and decides which chips are functionally correct. The IP owner also generates keys to unlock the passing ICs. Note that these keys are a function of the TRN generated in Step 1 and the random number generated in step 5 and are therefore unique to each IC.
9) A single message containing the ECIDs of the passing die and their associated keys are sent to the assembly. (Note that the IP owner can limit the number of keys sent out, thus preventing overproduction).
10) The assembly/distributor applies the keys sent by the IP owner (stores it in another OTP) to unlock functional block of the passing ICs. The key for each IC is different. The ICs that are still locked are useless to the assembly and are clearly non-functional, thereby making them easy to detect if inserted into the supply chain.

The main feature of the proposed CSST that separates it from the original SST is that only a single non-blocking session of communication is required between the foundry/assembly and the IP owner and vice versa.



(a)



(b)

Fig. 4. CSST: Modified (a) Functional-locking block and (b) Scan-locking block to remove the communication barrier.

A. Modified Functional-Locking Block in CSST

The functional-locking block is used to lock the functionality of an IC to prevent IC piracy and out-of-spec or defective counterfeits. Basically, an IC that is locked will operate very differently than its supposed to, thereby making it easy to spot in the supply chain. Figure 4 (a) shows the functional-locking block in the proposed CSST. The functional-locking block consists of an XOR_F mask, which is a series of m 3-input randomly inserted XORs, a TRNG, and two OTPs (OTP1 and OTP2). The XOR_Fs are only inserted into non-critical paths in order to avoid timing violations. The TRNG exploits the intrinsic random noise and process variations in a circuit to generate a true random number (TRN), unique to each IC. The TRN is stored in OTP1 so that it remains constant over the IC's lifetime. The structure and operation of a TRNG can be found in the literature (e.g., [16], [17]). The TRN is divided into three parts: T_{F1}, T_{F2}, and T_S. T_{F1} and T_{F2} are used to lock the IC functionality via XOR_F. An XOR acts like a buffer when its other input experiences '0' and acts like an inverter with '1'. T_S is used to control a scrambling block in the modified scan-locking block (see Section III-B). A 3-input XOR_F is used to send the value of the circuit path directly or inversely for functional locking. IN0 and IN1 are connected directly to the circuit path and T_{F1}, respectively. OTP2 is initially set to all 0s or all 1s, which is known to the IP owner. The T_{F2} and the output of OTP2 are XORed and connected to IN2. Initially, the value of OTP2 is all 0s or all 1s. The contents of OTP2 are XORed with T_{F2}. Hence, IN2 receives either T_{F2} or T'_{F2} depending on all 0s or all 1s in OTP2. Depending on T_{F1} and T_{F2}, the $XORF$s act like inverters or buffers. IN1 and IN2 are made different so the IC remains locked. To unlock the IC, the IP owner sends an $FKEY$ in such a way that IN1 and IN2 receive the same value and the $XORF$s become transparent. Only $T_{OTP2} = FKEY = T_{F1} \oplus T_{F2}$ satisfies this condition. The $FKEY$ can only be generated by the IP owner, who knows that the TRN is unique for each chip and it does not reveal any information about the TRN. Note that, in this design, one can trade-off OTP size for security by (1) using smaller k-length T_{F1}, T_{F2} and (2) using OTP2 to broadcast m XOR_Fs p times such that $m = p * k$ where p is an integer.

B. Modified Scan-Locking Block in the CSST

The scan-locking block is used to perturb the test response so that an outsider cannot determine the true test response from even an unlocked IC. Figure 4 (b) shows the scan-locking block and its mechanism. The yellow blocks represent the scan chain of the design and the test structure commonly used in practice while the rest are required to implement the CSST. Test data compression is required to overcome the limitations of the automatic test equipment (ATE). The outputs of the scan chains are scrambled through a scrambling block (SB) in order to perturb the functionally locked/unlocked response. The output of the SB is sent through the SO-XOR blocks for further perturbation. Scrambling blocks are an essential component in telecommunication and microprocessor communication [18]. The scrambling block can be completely shuffled or partially shuffled. The complete scrambling block (CSB) is designed such that all inputs to the block can potentially go to any output pin received by a compaction circuit. On the other hand, a partial scrambling block (PSB) is designed in such a way that an input to the scrambling block is connected with only N_{SB} different outputs. The non-blocking crossbar switch [18] is a strong candidate for the scrambling block and can be designed with pass transistors or transmission gates.

The security strength depends on the type of shuffling block. A CSB will provide maximum security at a higher cost. N_{SB} can be tuned to accommodate concerns with area overhead and security strength. The SB's controlling unit (SBCU) ensures that all the inputs to the SB, either the CSB or the PSB, are seen at the output. The logic of the SBCU depends on the PSB or CSB structure and the number of scan chains (N_{SC}). The output of the LFSR, which controls the SBCU, changes in each clock cycle and depends on the initial seed, T_{SIP}=$T_S \oplus T_{IP}$, which is only known by the IP owner. T_{IP} is the value stored in OTP3. In the foundry, T_{IP} is set to all 0s or all 1s. But, in assembly, the IP owner sends a random number, R_{IP}, independent of the IC and the TRN value. The initial seed of the LFSR, T_{SIP}, differs for the assembly and foundry of the same IC; hence, the SBCU performs different scrambling (see more in Section III). The output of the CSB/PSB is sent through an SO-XOR block to add another layer of security. The SB has to be ready before the intermediate output of the scan chain, SO_{IN}, is ready. In order to avoid a timing failure, the LFSR can be activated in the negative clock edge so that the SB is ready and SO_{IN} can pass through it.

The SO-XOR block is controlled by T_{SIP}. Depending on the second input of XOR in the SO-XOR block, the output of the SB flips or goes transparent and is only known by the IP owner (as T_S is revealed to the IP owner). The SB and the SO-XOR block make it impossible for untrusted fabs and untrusted assemblies to determine the correct output responses.

SB after Compaction Unit : The size of the scan chains increases with gate-counts and flip-flops, but ATE only has a limited number of channels. The CSB requires a large area to support all the scan chains in order to get a high-quality, perturbed response. The CSB provides the best security level but requires a large area. In order to reduce costs, the CSB can be placed after the compactor. An $r:1$ compactor reduces r^2 times area for the SB.

C. Advantages of CSST over SST

1) Communication Flow: Figure 3 shows the complete communication flow between the IP owner and the foundry/assembly in the CSST. Basically, the fab/assembly collects all the test responses from a wafer/die under testing: the ECIDs (which include the coordinates of the die on the wafer), the encrypted TRNs ($E(TRN)$s), and the compaction signature outputs (SIG_{FAB} for foundry and SIG_{ASMB} for assembly). The foundry/assembly sends the collected ECIDs, corresponding signatures (SIG_{FAB}), and encrypted TRN values all at once. In the original SST, both the foundry and assembly *need to wait for a TKEY to perform the testing on each die.* The TKEY is unique for each IC, as it's TRN dependent. In the original SST, the foundry sends the TRN to the IP owner. The IP owner receives the TRN and modifies it, which is only known to the IP owner. The encrypted, modified TRN (TRN-mod) is sent to the foundry again, which is the TKEY for that particular IC. The foundry applies the TKEY and test vectors in order to test the IC. After testing, the foundry sends the perturbed response (a function of TRN and $TKEY$) to the IP owner. The same procedure is performed in the testing of other ICs and is also done after packaging by assembly. The IP owner checks the signature and returns an $FKEY$ to the assembly or authorized distributor if the die is passed. This multiple-session complex (blocking) communication flow between the IP owner and foundry/assembly is the main limitation of the original SST. The proposed CSST reduces the test time dramatically by using a very simple communication flow, which employs new functional-locking and scan-locking blocks. This makes the CSST more easily adopted in practice.

2) Security Enhancements: The CSST also has some additional security features, such as tunable locking blocks. The functional locking mechanism (a function of TRN value and XOR_F locations) is similar for both versions of the SST. The scan-locking mechanism of the original SST offers one layer of security, i.e., the XORs at the output of the scan chains with the test key, $TKEY$. But the scan-locking mechanism of the original SST also inverts the input to scan chains in order to prevent attacks in an unlocked IC. By contrast, the CSST uses two layers of security for both locked and unlocked ICs. The first layer consists of the SB that permutes the output of the scan chains in a different way for every test pattern based on the LFSR seed, T_{SIP}. The second layer contains XORs between MISR and the output of the original scan chains as in the original SST. Designing the SB to minimize cost will be investigated in future work. The communication sessions required for the CSST are slightly less than they are for the original SST. In the next section, we will compare the two in terms of security, area overhead, and attacks.

IV. Experimental Results and Analysis

Security Analysis : In order to verify the effectiveness and strength of the proposed CSST, we simulated two different benchmarks (40 ICs for each), using Synopsys's logic verification tool VCS [21]. The hamming distance (HD) is a popular metric for analyzing security strength. The effectiveness of the scan-locking block is analyzed since the perturbation of the functional-locking block alone on the CSST is same as it

TABLE I

HAMMING DISTANCE COMPARISON FOR DIFFERENT NUMBER OF XORS IN SO-XOR BLOCK.

N_{SO-XOR} (10% to 50% of N_{SC})		SST		CSST			
				SB before compaction		CSB after compaction	
s38417	b19	s38417	b19	s38417	b19	s38417	b19
1	5	9.06	9.91	29.29	38.18	42.87	43.76
2	10	19.66	13.61	40.01	43.69	46.41	48.12
3	15	22.89	20.54	48.73	49.31	47.12	47.78
4	20	25.79	26.66	47.44	49.31	49.31	48.89
5	25	36.36	36.43	50.03	49.31	50.03	50.00
6	30	46.46	39.87	45.63	50.00	49.31	48.89
7	35	47.44	43.69	47.44	49.31	50.03	50.00
8	40	49.31	48.89	50.03	50.00	50.03	50.00

TABLE II

HAMMING DISTANCE ANALYSIS FOR DIFFERENT SCRAMBLING BLOCKS IN CSST WITH N_{SO-XOR}=$N_{SC}/2$.

Benchmark	s38417				b19			
N_{SB}	2	3	5	10	5	10	25	50
%HD (Foundry)	42.04	44.59	50.03	50.03	48.07	48.84	49.31	50.00
%HD (Foundry, Assembly)	28.17	36.47	39.82	42.04	31.84	37.81	44.59	48.41

TABLE III

%HD ANALYSIS OF CSST PLACING CSB AT POST-COMPACTION FOR DIFFERENT COMPACTION RATION, R, OF A COMPACTOR N_{SO-XOR}=$N_{SC}/2$.

Benchmark	s38417		b19	
Compaction ratio, r	2	5	5	10
%HD (Foundry)	42.04	44.59	48.41	49.04
%HD (Foundry, Assembly)	28.17	36.47	39.82	42.04

is in the original SST. The average %HD between the actual response and the captured response from the modified scan-locking block is presented in Table I and II. A $\sim 50\%$ HD is hard to predict and represents high security, while $\sim 0\%$ or $\sim 100\%$ HD is easy to predict. The IC was a synthesized implementation of the ISCAS89 benchmark s38417 and ITC'99 benchmark b19. The result shows that the modified-scan-locking block contributes well to perturbing the actual response, regardless of the ICs size. s38417 and b19 have 10 and 50 scan chains, respectively. The functional-locking mechanism has the same hamming distance as the original SST. Security is enhanced in the CSST, both by the shuffling block and the SO-XOR block. Table I shows the effectiveness of the SB in the CSST. The result shows that for $N_{SB} = N_{SC}/2$, there is a huge shift in the hamming distance to the ideal value (50%), which demonstrates that the CSST possesses a higher level of security than the original SST. The result also shows that the original SST requires more XORs in the SO-XOR block to obtain 50% HD than the CSST because of the additional shuffling block. The location of the SB impacts the security of the whole system. The SB can be placed between the scan chain and the compaction circuit or after the compaction unit (post-compaction) to reduce the area overhead. The result shows that moving the SB after compaction unite provides better hamming distance due to the effectiveness of the shuffling block. Table III shows the effect of placing the SB after compaction unit with different compact ratio of MISR (compactor). Placing the SB after compaction unit offers high-quality security with lower area overhead. The total number of XORs, N_{SO-XOR}, in the SO-XOR block were varied to understand the effectiveness of the SO-XOR block. The result shows that N_{SO-XOR} is an important parameter and needs to be large in the old SST. The result shows that different N_{SO-XOR} values are required for different benchmarks. On the other hand, the CSST assures high security with the shuffling block and requires a small N_{SO-XOR} value when compared to the old SST. Table II shows the effectiveness of the SB and R_{IP}. The results show that N_{SB}=10% N_{SC} can achieve an ideal hamming distance. The same die gives different responses from foundry to assembly because the random number, R_{IP}, provided by the IP owner scrambles the response differently. The last row of Table II shows that the foundry and assembly possess significant hamming distance between responses for the same IC. The SB usually takes up a large area in the CSST, but high security can be achieved by tuning the SO-XOR block and the SB. Tables I and II show that activating both the PSB and the partial SO-XOR block can be used to achieve the desired security with the lowest area. The result shows that N_{SB}=50% N_{SC} and N_{SO-XOR}= 50% of other N_{SC} ensures maximum security. The simulation results also show that the CSST offers less standard deviation in HD than the original SST (not shown in the tables for brevity).

TABLE IV

AREA OVERHEAD FOR A 10M GATES DESIGN WITH $k = 1024$ AND $N_{SC} = 1000$.

	SST	CSST			CSB after compaction	
m		$N_{SB} = 10$	$N_{SB} = 100$	$N_{SB} = 1000$	$r = 50$	$r = 100$
1024	0.0222%	0.0366%	0.1365%	1.1355%	0.0244%	0.0366%
2048	0.0233%	0.0377%	0.1376%	1.0366%	0.0255%	0.0377%
5192	0.0266%	0.04%	0.1399%	1.1389%	0.0289%	0.04%
10240	0.0322%	0.0466%	0.1576%	1.5762%	0.0344%	0.0466%

Area Overhead : We calculated the area overhead as follows: RSA can take k-bit input and give k-bit output, provided that the length of the public key, K_{pub}, is also k bits. RSA encryption's throughput and speed is higher than RSA decryption and requires less area overhead [19]. The modified functional-locking block requires m XOR_Fs, where $m = p * k$ and p is an integer. Each of k-bit T_{F1} and k-bit T_{F2}, and k-bit T_{OTP2} can be broadcast p times in order to connect all m XOR_Fs. The ring oscillator based TRNG is easy to implement and does not need to add additional circuitry as k increases. The size of the LFSR depends on the size of the SB, and the size of the SB depends on N_{SB} and the total number of scan chains, N_{SC}. N_{SB}=N_{SC} for a CSB and $N_{SB} < N_{SC}$ for a PSB. The size of the SBCU depends on N_{SC} and the structure of the SB. Table IV shows the area overhead comparison between the SST and the proposed CSST. The result shows that the CSST is a strong function of N_{SB} and N_{SC}. The size of the SB, the LFSR, and the SBCU depend on the number of scan chains, N_{SC}. The main cost of the modified scan-locking block is the SB and the SBCU. The area overhead can be reduced by placing the CSB after the compaction unit. The area overhead for placing the CSB after compaction unit depends on the compaction ratio of a compactor. The result shows that the area overhead is reduced significantly if the CSB is placed after the compactor. On the other hand, the XOR is inserted in no critical path and, hence, no timing violation is observed.

Attacks Analysis : The CSST is resilient to all possible attacks discussed in the context of the original SST [13].

1) A *circumvent attack* is extremely challenging in the CSST. Knowledge of the location of XOR_Fs in the modified functionallocking block will not reveal any information, as the foundry does not have any information

about T_{F1} and T_{F2}.

2) A *tampering attack* requires re-routing the outputs from the TRNG so that they go directly to the output of RSA, bypassing the RSA decryption needed to activate the IC, which would be prohibitively expensive. Bypassing RSA is difficult, as the untrusted foundry does not have any knowledge about the length and position of T_{F1}, T_{F2}, or T_S, and the $FKEY$ is also different than the TRN.
3) A *removal attack* is more difficult than it is in the previous SST. Removing the XOR gates whose output is connected to a flip-flop is a fruitless attack now because of the SB in the modified scan-locking block.
4) Testing *unlocked ICs* to get the correct response is also more difficult than it is in the previous SST because of the SB and the fact that the FKEY does not leak any information to the adversary.
5) In addition, a *flush and shifting attack* is also impossible due to the SB in the scan chain. The flush test is performed to find any defects in a scan chain. In the flush test, a selected flush pattern (e.g. 11001100 or 111111, or 000000) is shifted all the way through the scan chain input with the expectation that the same pattern will arrive at the output. An attacker might try to shift in all 0s or all 1s to obtain the functionality of a scan-locking block (specifically, T_S), but the CSB and PSB make it nearly impossible. Note that the prior version of the SST is susceptible to this attack.
6) The proposed CSST is also resilient to *graph isomorphism* attacka. [9] described that this potential attack can reveal the locking mechanism. But, in the CSST, an adversary does not know the functionality of the logic network, as he/she does not get an exact test response from testing because of the scan-locking block. Another potential attack is that the untrusted foundry will send random encrypted TRN to get an idea of the correct response. Or an adversary can send the same TRN for different ICs. However, whenever the IP owner receives the wrong TRN, it won't pass the IC. Several attempts could be viewed as a low yield and hence harm the reputation of the foundry/assembly.
7) An adversary might change the OTP value to all 0s or to all 1s to make the XOR_F transparent. To prevent this attack, an XOR_F block is implemented by inserting both XOR and XNOR randomly. In order to prevent such attacks on scan chain, the LFSR is also designed in such a way that, for all 0s, it does not output all 0s, and the SO-XOR block might be built using both XOR and XNOR.

Summary of the CSSTs Advantages: The simplicity of communication flow is the main strength of the CSST. In addition, the CSST is more resistant to attacks most notably the flush and shift attacks - than the SST because of its modified scan-locking block. The CSST is also more flexible with respect to security-area trade off due to the different tunable parameters in the modified scan-locking block. The CSST offers less on-chip timing overhead than EPIC [7] and the original SST [13] because there is no RSA decryption block inside the chip.

V. Conclusion

We have presented a modified Secure Split-Test called the Connecticut Secure Split-Test (CSST) that prevents IC piracy completely using a very simple communication model flow between the IP owner and the foundry/assembly. We believe that this will make the CSST more easily acceptable to both foundries/assemblies and IP owners. In addition, the CSST offers better security and more tunable parameters to scale the security of testing compared to the previous SST. In future work, we will analyze ways of inserting XOR for functional locking in order to achieve the highest security with least number of XORs.

VI. Acknowledgment

This work was supported in part by the National Science Foundation under grant CNS-0844995 and Army Research Office under grant 57958CS.

References

[1] DIGITIMES, "Trends in the global IC design service market," http://www.digitimes.com/news/a20120313RS400.html?chid=2
[2] U. Guin et al., " Counterfeit Integrated Circuits: Detection, Avoidance, and the Challenges Ahead," Journal of Electronic Testing, vol. 30, pp. 9-23, 2014.
[3] Y. Alkabani et al., "Remote Activation of ICs for Piracy Prevention and Digital Right Management," in IEEE/ACM International Conference on Computer-Aided Design, pp. 674-677, 2007.
[4] F. Koushanfar, "Provably Secure Active IC Metering Techniques for Piracy Avoidance and Digital Rights Management," IEEE Transactions on Information Forensics and Security, vol. 7, no. 1, pp. 51-63, 2012.
[5] R. Chakraborty and S. Bhunia, "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 28, no. 10, pp. 1493-1502, 2009.
[6] J. Rajendran et al., "Security analysis of logic obfuscation," in the Proc. of IEEE/ACM Design Automation Conference, pp. 83-89, 2012.
[7] J.A. Roy et al., "EPIC: Ending Piracy of Integrated Circuits," in proc. Design, Automation and Test in Europe, pp.1069-1074, 2008.
[8] A. Baumgarten et al., "Preventing IC Piracy Using Reconfigurable Logic Barriers," IEEE Design and Test of Comp., vol. 27, pp. 66-75, 2010.
[9] B. Liu and B. Wang, "Embedded Reconfigurable Logic for ASIC Design Obfuscation against Supply Chain Attacks," Design, Automation and Test in Europe Conference and Exhibition, pp. 1-6, 2014.
[10] R. S. Chakraborty and S. Bhunia, "Hardware Protection and Authentication through Netlist Level Obfuscation," In Proc. IEEE Intl. Conf. Computer-Aided Design, pp 674-677, 2008.
[11] Md. Tauhidur Rahman et al., "ARO-PUF: An Aging-Resistant Ring Oscillator PUF Design," Design, Automation and Test in Europe Conference and Exhibition (DATE), vol., no., pp.1, 6, 24-28, 2014.
[12] K. Xiao et al., "Bit selection algorithm suitable for high-volume production of SRAM-PUF," IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 101-106, 2014.
[13] G. Contreras, M. T. Rahman, and M. Tehranipoor, "Secure Split-Test for Preventing IC Piracy by Untrusted Foundry and Assembly," in Int. Symposium on Defect and Fault Tolerance in VLSI Systems, 2013.
[14] M. Tehranipoor et al., " Trustworthy Hardware: Trojan Detection Solutions and Design-for-Trust Challenges," IEEE Computer Magazine, Vol. 44, Issue 7, pp. 66-74, 2011.
[15] E. Love, Y. Jin, and Y. Makris, " Enhancing security via provably trustworthy hardware intellectual property," IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), 2011.
[16] Berk Sunar, "True Random Number Generators for Cryptography," Cryptographic Engineering, pp. 55-73, 2009.
[17] Md. Tauhidur Rahman et al., "TI-TRNG: Technology Independent True Random Number Generator," In Proceedings of the 51st Annual Design Automation Conference on Design Automation Conference (DAC), Article 179, pp. 1-6, 2014.
[18] Y. Tamir and H.C. Chi, "Symmetric Crossbar Arbiters for VLSI Communication Switches," IEEE Transactions on Parallel and Distributed Systems, vol. 4, no. 1, pp. 13-27, 1993.
[19] C. McIvor et al., "Fast Montgomery Modular Multiplication and RSA Cryptographic Processor Architectures," Proc. 37th IEEE Comp. Society Asilomar Conf. on Sig., Sys. and Comp., pp. 379-384, 2003.
[20] Md. Tauhidur Rahman et al., "CSST: An Efficient Secure Split-Test for Preventing IC Piracy," In IEEE North Atlantic Test Workshop, 2014.
[21] Synopsys VCS: Functional Verification Tool, Synopsys; version 2010.03.