CLIP: Circuit Level IC Protection Through Direct Injection of Process Variations

W. Paul Griffin, Anand Raghunathan, *Fellow, IEEE*, and Kaushik Roy, *Fellow, IEEE*

***Abstract*—The disaggregation of the semiconductor design and manufacturing process has resulted in integrated circuit (IC) piracy becoming an important concern to the semiconductor industry. To address this concern, we present a method for achieving robust IC protection at the circuit level through direct injection of process variations. In the proposed approach, the circuit is enhanced by including *process variation (PV) sensors* and modifying the design during synthesis to inject the outputs of the PV sensors into the logic at carefully selected nodes. As a result, each fabricated IC is rendered inoperative unless a *unique per-chip unlocking key* is applied. After fabrication, the response of each chip to specially generated test vectors is used to construct the correct per-chip unlocking key. We propose a methodology to automatically modify circuits by identifying pairs of injection and correction points, while avoiding delay penalty and minimizing area overheads. We propose the use of a cryptographic preprocessor to separate the internal key used from the external unlocking key, further enhancing the resistance of the proposed approach against several attacks. Our methodology is scalable to the key size and requires only a small area overhead to achieve reasonable security levels (e.g., 7% for 64-bit keys in a 8 k gate design). We analyze the security of the proposed technique under several attack scenarios and believe that it offers robust protection against a wide range of attacks.**

***Index Terms*—Hardware security, integrated circuit (IC) piracy, process variations, reverse engineering.**

I. Introduction

THE increasing complexity and cost of modern integrated design and fabrication has led to significant changes in the semiconductor industry. To cope with these trends, the industry has restructured itself to reduce expenses, facilitate knowledge sharing, and globalize operations. Companies that complete an entire design in house are a rarity—many designs are produced through a process involving Intellectual Property (IP) resellers, system on a chip (SOC) design houses, and fabrication plants that span multiple countries. This dispersion of resources across corporate and legislative boundaries naturally raises significant concerns about piracy.

IP providers are concerned that their designs will be leaked, resold, or overused by design houses; design houses are worried about handing masks to foundries they cannot monitor; and foundries are worried that their reputation and revenues could be jeopardized by a rogue employee. Given the large work force that goes into the design and manufacturing of an IC, it should come as no surprise that an estimated 80% of piracy is as a result of internal design theft [1].

ICs that are outsourced for fabrication are especially vulnerable to piracy from overproduction and mask theft. For example, it is conceivable that a dishonest manufacturing plant could create more chips than ordered and sell the additional chips at a lower cost, subverting the profits of the legitimate owner.

Legislative protection for chip designs is not sufficient to address piracy concerns. The U.S. Semiconductor Chip Protection Act [2] (and similar international legislation [3]–[5]) gives the mask work owner exclusive rights to manufacture and sell original designs. However, such legislation is not present in all parts of the world. Furthermore, mask reverse engineering is not forbidden even under existing laws: provisions are included that allow the sale of a design created from knowledge gained through reverse engineering, as long as the resultant work can be considered original. As a result, for many companies, reverse engineering competitors' chips is a routine business practice, and is further simplified by companies that provide tools and services for IC reverse engineering [6]. While reverse engineering an IC does incur some cost, layout, and netlist extraction is still far cheaper than design costs.

Due to growing concerns across the IC industry, a number of approaches have been proposed to help prevent piracy and design theft. For IP cores, these range from encryption of the design to watermarking and functional obfuscation [7]–[9]. For fabricated ICs, both passive schemes such as unique chip identifiers and active schemes such as chip locking or metering have been proposed [10]–[14]. Process variations, which are an inevitable consequence of scaled manufacturing technologies, are utilized to realize some of the IC protection schemes. While these techniques have made promising advances, significant challenges remain. Revealing the mask to a foundry means that one must be concerned with modifications of the mask that can disable the protection schemes. Specifically, techniques that have a single "point of vulnerability" are especially susceptible to mask modifications.

In this paper, we propose a scalable method to achieve circuit-level IC protection (CLIP) through direct injection of

Manuscript received June 22, 2010; revised October 08, 2010 and January 26, 2011; accepted February 21, 2011. Date of publication May 05, 2011; date of current version April 06, 2012.

W. P. Griffin is with Department of Electrical Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: wgriffin@ecn.purdue.edu).

A. Raghunathan is with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-2035 USA (e-mail: raghunathan@purdue.edu).

K. Roy is with School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-1285 USA (e-mail: kaushik@ecn.purdue.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2011.2135868

process variations. Our approach exploits process variations similar to other active schemes; however, it differs significantly in the manner in which they are utilized. The circuit is enhanced during design by "injecting" the outputs of digital *process variation (PV) sensors* into the logic in a *distributed* manner at carefully selected injection nodes, rendering the IC inoperative upon fabrication. A corresponding set of correction nodes is identified and utilized to reverse the effects of process variation injection when a *unique per-chip unlocking key* is applied.[1] Uniqueness of the unlocking key is realized by requiring it to be based on process variation (PV) sensor readings. As a further level of defense, we utilize a cryptographic preprocessor to transform the unlocking key into the internal key used to reverse the effects of the PV sensors. Knowledge of the design modifications is utilized to generate special test vectors that propagate the PV sensor outputs to the circuit's primary outputs. After manufacturing, the responses of each IC to these test vectors are used to compute its unique unlocking key. We analyze the security of the proposed scheme under a range of external and intrusive attacks. The significant contributions of our work are as follows.

- We demonstrate how to utilize direct and distributed injection of process variations to realize chip locking with unique per-chip unlocking keys. Unlike previous schemes that use unique external unlocking keys that get transformed to a fixed internal key, direct injection ensures that keys remain unique per-chip at all levels.
- The distributed insertion of process variation sensors and circuit elements implies that there is no single point-of-attack that can compromise the proposed technique.
- The injection and correction nodes can be arbitrarily separated from each other and merged into the functional logic during synthesis, increasing the difficulty in reverse engineering and identification.
- We achieve scalable security by allowing the designer to increase the length of the unlocking key at a reasonable increase in hardware overhead.
- No sensitive information is exposed from the chip by means of a scan chain or similar mechanisms. Rather, PV sensor information is obtained through logic paths upon application of specific test vectors, and the responses are used to compute the unique unlocking key.

In Section II, we present relevant background to our work. In Section III, we demonstrate a systematic methodology to enhance any given circuit for CLIP. The core logic modification to allow for injection and correction of process variations is presented in Section III-A. Section III-B presents a process variation sensor that exploits random dopant fluctuations, together with an explanation for why we believe such a sensor is feasible. We examine the test and unlocking procedure for proper IC operation in Section III-C, and explain the necessity for a preprocessor that forbids direct application of a key to the logic. We demonstrate the proposed technique on a range of benchmarks in Section IV, followed by an analysis of various attacks in Section V. We conclude our work in Section VI.

II. Background and Related Work

A significant body of work has addressed the problems of IP and IC protection, and we summarize the major efforts below.

The business model of IP core providers squarely depends on the ability to ensure that their cores are kept secure. IP cores can be provided from an IP vendor to a design house in an encrypted form [15], [16]. While this adds some measure of security, the implementations of IP cores are exposed (as netlists or layouts) at later stages of the design flow. A commonly proposed IP protection technique is watermarking, wherein a watermark or signature is embedded into the design that can be used for *a posteriori* detection and enforcement. Watermarks can be created through the use of physical positioning to create observable features [7], added constraints to optimization problems during synthesis [8], [9], or using extraneous functionality such as unused transitions in a state transition graph [17].

Field-programmable gate-array (FPGA)-based designs are particularly vulnerable to piracy and exploitation since their designs must be transmitted via a bitstream to the FPGA on system power-on. To counter this issue, physically unclonable functions (PUFs) have been considered as a viable method for locking a given bitstream to a particular FPGA. PUFs, first proposed by [18], produce an irreplicable challenge-response mechanism driven by process variations. PUFs have several advantages over the use of nonvolatile memory: they do not require active tamper protection, do not require additional masks during manufacture, and do not require programming by a trusted party to achieve their uniqueness [19]. FPGA-based PUF designs rely on measurements taken from delay and memory elements [20], [21], and silicon-based PUFs have been proposed based on delay, memory, and capacitor-based sensors [22]–[24].

IC protection schemes aim to protect the design or mask after it is released to the foundry, as well as fabricated instances of the IC that are deployed in end systems. They can be classified into several categories. *Passive schemes*, such as unique chip IDs or PUF-based fingerprints that are registered into a database [25], rely on a posteriori detection rather than pre-emption of piracy.

Recent work has focused largely on *active protection schemes*, where ICs are inherently manufactured in a locked state pending the application of an unlocking key. These schemes can be used to implement "metering," wherein the legitimate owner of the design releases unique unlocking keys for each manufactured instance. Recently, a class of active protection schemes have been studied that modify the finite-state machine (FSM) representation of the circuit in order to introduce protection. Reference [11] proposed adding states to a FSM to connect a PUF and key, while [12] proposed introduction of a deterministic FSM that requires an activation sequence supplied at bootup through the primary inputs.

In [13] and [14], key exchange protocols were proposed to help protect the unlocking of an individual IC. These protocols relied on PUF-driven fingerprints at the key exchange level to require a unique external unlocking key. Some vulnerabilities and improvements to these protocols were presented in [26]. Similar protocols have also been implemented commercially [27].

[1] Uniqueness is desirable since a key that is reverse-engineered or leaked is of no use beyond the single instance that it unlocks.

TABLE I
COMPARISON AGAINST ACTIVE PROTECTION TECHNIQUES

	Remote Activation [11]	HW Protection [12]	EPIC [13]	IC Activation [14]	CLIP
In-Field Reauthentication	✓		✓	✓	
Unique Unlocking Key	✓		✓	✓	✓
Unique Internal Key	✓				✓
Single point of vulnerability	×	×	×	×	
Requires FSM Representation	×				

While piracy prevention may be related with security mechanisms used for IC identification or user authentication, these mechanisms are built to serve the end user (protection starts primarily after the IC is incorporated into an end system), whereas piracy prevention is intended to aid the design house (protection starts when the IC is fabricated). If the objective is piracy prevention, one should not expect an end user to be a willing active participant—upon purchase, they expect to receive an unrestricted product; they do not wish to contact the manufacturer nor repeatedly authenticate themselves to unlock their chip again.

The objective of CLIP is IC piracy prevention, and it falls under the category of active protection schemes. Therefore, we closely compare it with other approaches that fall under this category. Existing active protection schemes may be effective at implementing protection against external attacks, but significant improvements can still be made on protecting against attacks with a knowledge of the internal structure and mask. A unique external unlocking key is achievable by previous schemes, but key transformations and combination with internal values culminate in: 1) supplying a fixed, common internal key to the logic core to unlock it's operation or 2) a single signal enabling or disabling the operation of the design. These limitations lead to simple attacks. For example, probing at the location of the fixed internal key could reveal the fixed internal key, followed by mask modification to hardwire the internal key. Alternatively, the protection scheme could be permanently disabled through a single mask modification.

The specific differences between CLIP and the known active protection schemes are summarized in Table I. The first row indicates that CLIP focuses on piracy prevention, and does not target in-field reauthentication. The second and third rows suggest that CLIP utilizes not only unique unlocking keys, but also unique keys internally. The only other technique that does this is [11]. The fourth row implies that all of the other techniques have a single point of vulnerability. Finally, we point out that [11] is based on explicit FSM modification, which is not scalable since construction of the FSM is not practical for even moderately sized circuits.

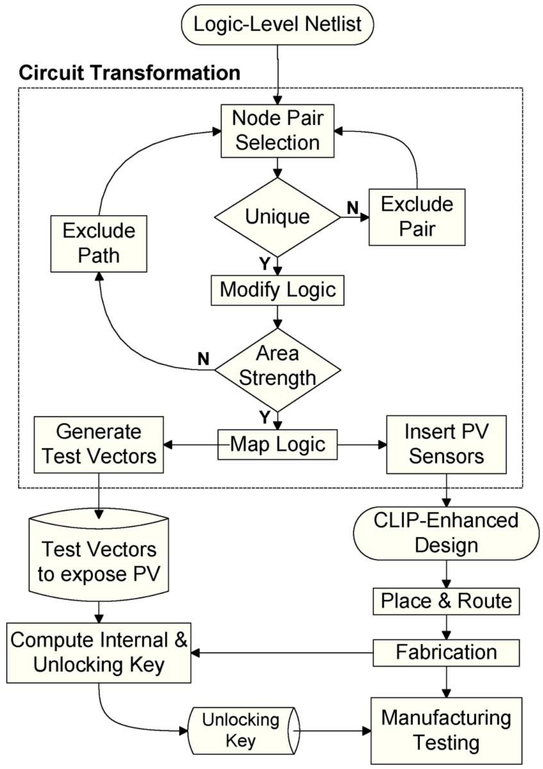


Fig. 1. Design flow for CLIP.

With CLIP, we achieve enhanced security with respect to previous approaches. We inject the key and process variations into the logic directly and in a distributed manner, creating a system wherein no static, common internal key exists prior (or even inside of) the logic core. The insertion of sensors directly into the logic minimizes probing opportunities, while distributing any potential vulnerabilities across the mask. The distributed nature of our scheme also allows for high scalability—we achieve a linear relationship between the key length and the added logic, giving designers tight control on the hardware overhead. We present a comprehensive integrated synthesis methodology for selecting the injection and correction points so as to avoid delay overheads whenever possible, while maximizing functional impact and ensuring the ability to recover PV sensor values through the logic paths in the circuit.

III. CIRCUIT-LEVEL IC PROTECTION

Fig. 1 contains a flowchart applying our methodology for CLIP. Starting with a logic-level netlist, we analyze and perform transformations on the logic, producing a CLIP-enhanced design [see Fig. 2(a)]. The logic core modification includes selection of suitable logic regions [see Fig. 2(b) and (c)], logic modification for injection and correction [see Fig. 2(d)], and the addition of digital PV sensors [see 2(e)]. Based on the circuit transformations, test vectors are generated to expose the process variations at the circuit outputs, and after fabrication, the test vector responses are used to compute the unlocking key for each manufactured chip.

To integrate the digital sensor outputs and key into the logic, a pair of nodes in the circuit are selected as potential locations for PV injection and key correction based on predetermined heuristics. For selected node pairs, the circuit is modified to allow for proper operation under the presence of injection. Nodes that lie in the path between the selection and correction nodes are excluded from further analysis, and the process is repeated until

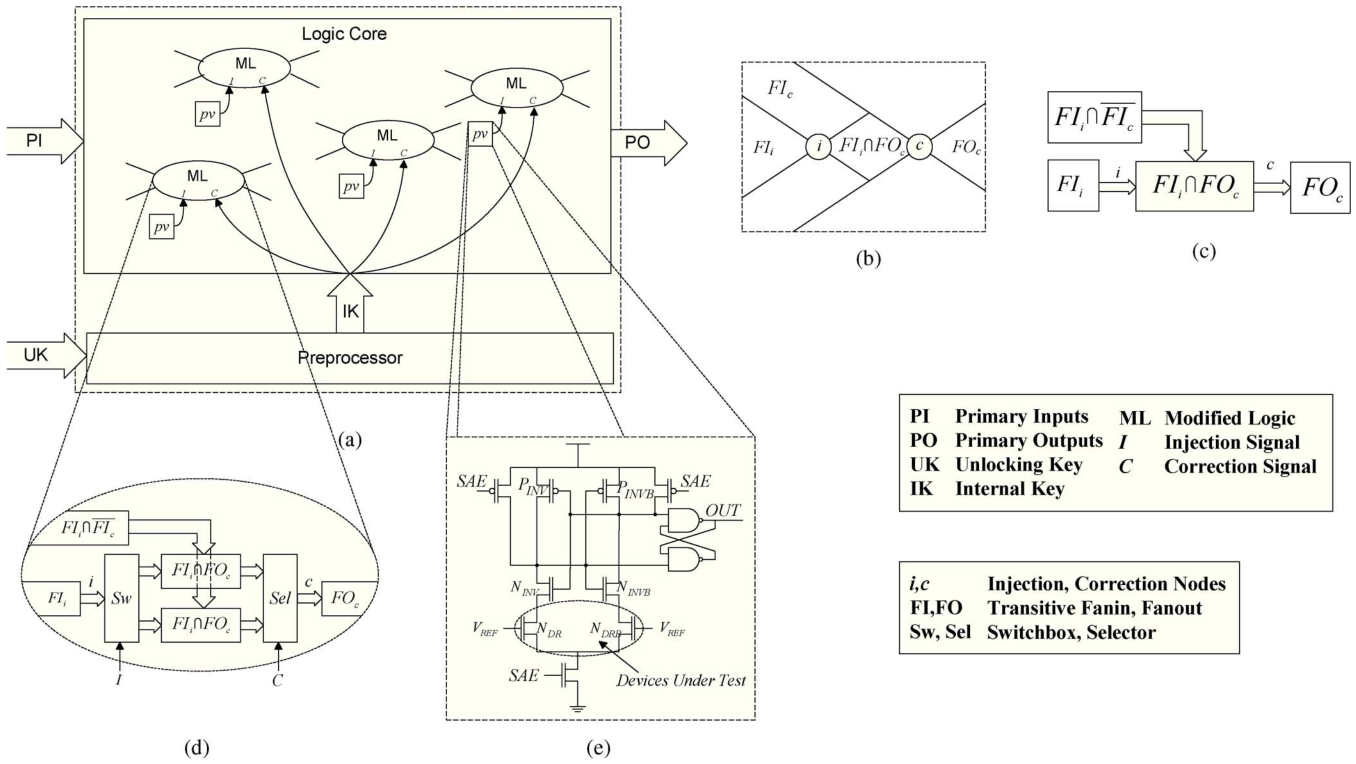


Fig. 2. Overview of CLIP. (a) CLIP-enhanced design with multiple logic modifications and PV sensors, (b) logic cones (fan-in and fan-outs) for a set of injection and correction nodes, (c) alternate representation as logic blocks, (d) modified logic with logic duplication, switchbox, and selector, (e) sense amplifier-based PV sensor. The nMOS devices under test operate in the subthreshold regime and have a low duty cycle to enhance their stability.

the desired unlocking key strength is achieved, hardware overhead limitations are reached, or the logic cannot fit any more nodes without increasing the critical path.

After logic synthesis, suitable PV sensor cells are added. To the circuit's logic, they only need to behave as black-box random number generators. These sensors can vary in construction, but ideally would be structured such that they are not easily distinguishable from surrounding logic.

To discover the unlocking key for the design, a set of test vectors are constructed that expose the PV sensor readings. While one can certainly use a more thorough test generation process, we developed a process which is fast and produces good results. To generate the test vectors, random vectors are applied to the primary inputs and propagated through the logic such that the logic's output is only dependent on the sensor values. While some sensor values can be immediately obtained from the outputs (if an output is directly dependent on a sensor reading), others require iteration through a subset of keys. The determined sensor values enable the corresponding key bits to be used as primary inputs for greater pruning in later stages. This reuse causes different test sets for each chip; however, the test set does not need to be recomputed from scratch since the test vectors are universal with respect to the key/process variation mapping.

Due to a trivial mapping between the internal key and the process variations, advanced circuit knowledge coupled with methods such as differential analysis [28] may allow an outsider to recover a single IC's key. However, if we can make the application of a particular internal key nontrivial, an outsider cannot defeat the methodology so easily. Instead of directly applying a key to the logic, an external, *unlocking key* is fed through a preprocessor to generate an *internal key* which directly feeds into the core logic. This preprocessor could be verification-based to, for example, prevent application of any key that does not fulfill checksum requirements, or it could be encryption-based and apply a secret transformation to the key.

A. Logic Modifications for CLIP

To perform logic modifications, the circuit is analyzed based on the logic cones that feed and are driven by the injection and correction nodes, and then transformed through a combination of logic duplication and additional circuitry [see Fig. 2(b)–(d)].

Since the circuit receives static random values from the sensors, the circuit's construction must allow for some means of correct operation for any given sensor reading; a key must exist for all potential sensor outputs. Under a trivial scenario, bits of the key and sensor readings could be combined before attachment into the circuit, and are subsequently combined into the existing logic via an obfuscating (logic-modifying) gate. While functional, each modification would be vulnerable to a single point of attack. Once a wire is tied to either $V_{\rm dd}$ or $V_{\rm ss}$, the modification is placed into a permanent and reproducible unlocked state. Instead, if we separate the key and PV connections, we can quickly realize a design that eliminates the fixed internal key and doubles the required number of mask modifications to reach the unlocked state without increasing the key size or the number of PV sensors.

To account for different key and PV values, the overlapping fanout and fanin of the injection and correction nodes, respec-

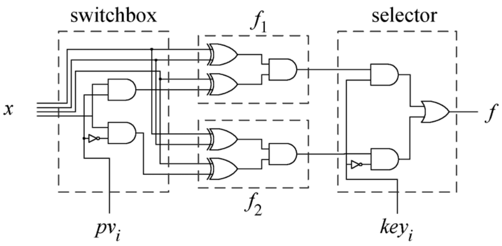


Fig. 3. Logic modifications for correctibility. Based on the PV sensor reading, the switchbox blocks value propagation to one of the duplicated functions (f_1, f_2) such that only one function is reliably correct. A key input, fed to the selector, determines which function's value to use.

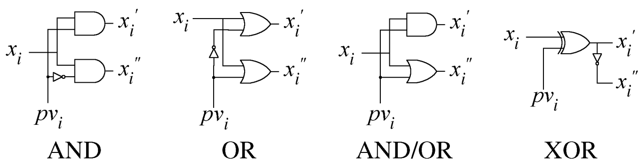


Fig. 4. Single-bit switchbox styles. The variety of options give a designer a greater opportunity to conceal the injection nodes from mask analysis.

tively, are duplicated to create a second logic block. The input path to the blocks is fed through a switchbox, which determines which circuit copy receives the originally intended data. After each copy performs an operation on the data, a selector is used with a corresponding internal unlocking key to choose the block that obtained the proper result. Fig. 3 presents an example circuit modified for correctibility. The switchbox uses the PV sensor output to block some of the data supplied to the duplicated logic (in our case, a 2-bit equality checker). The logic executes the two sets of data in parallel, and then a two-way multiplexer attached to the key decides which output to trust.

A single-bit input switchbox allows for a variety of implementations, including dual AND, dual OR, mixed AND/OR, XOR, and their complements (see Fig. 4). Dual AND and dual OR implementations allow the input signal to propagate to one of the copies of the duplicated logic, and always feeding a constant logic value (0 or 1) to the incorrect copies. To minimize area, a mixed AND/OR implementation can be used, as the side inputs to the gates allow propagation for complementary values. However, if an XOR implementation is used, switching activity is more likely to propagate to the outputs.

Multibit switchbox implementations can be implemented using any random logic, provided the switchbox follows one restriction: dependent on the switchbox side input, one output must always be correct while the other paths are incorrect. Functionally, this can be accomplished by splitting the subfunction's input signal into two paths and then inserting various styles of logic obfuscators (primitive gates that alter the output when enabled), driving each path's obfuscators by complementary signals. This allows for asymmetric implementations, which could potentially confuse side-channel attacks.

Custom implementation of a suitable selector is more constrained, but it does contain alternate options. With the typical two-level AND-OR MUX, the AND gates create a small unate region prior to recombination at the OR gate. As long as this unate property is preserved when the two paths reach the recombination gate, the selector can be reimagined by moving the location or changing the implementation of the unate-causing gates.

The placement and selection of the switchboxes and selectors is largely a heuristic problem, which, while issues such as area minimization, critical path avoidance, and testability should be a major consideration, allows for the addition of many additional constraints such as preserving internal node probabilities. While there is plenty of exploration space with respect to node placement, the following few simple observations pertaining to node selection can be made.

- **Single-Bit Components**. Switchboxes that affect more than one modification per output path, and selectors that recombine more than one signal have little impact on the device's security: additional logic is required without any increase in the key length. Single-bit components give us a minimal-area method for removing the fixed internal key.
- **Fan-out-Free Cones**. To minimize logic duplication, it would be ideal to only clone preexisting logic once. We can accomplish single duplication by restricting the node selection to when the correction node is at the tip of a fan-out-free cone that contains the injection node. If we follow this guideline, we can automatically avoid the use of multibit selectors.
- **Overlap Avoidance**. If the duplication region between the selector and the switchbox overlaps with another node pair's duplication region, some of the original logic will appear four times in the modified circuit. By using a node selection procedure that tracks duplicated gates or relocates the injection node, we can avoid overlapping multiple sets of node pairs.

B. Process Variation Sensors

While any process variation sensor that can return a static random value will function with our method, the sensor should be low overhead and have a consistent output. Sensors with low area are ideal, as many such sensors could be embedded in the circuitry. The sensor's output should also be invariant to operation-induced variations such as thermal effects. While a formal analysis of the behavior of PV sensors is outside the scope of this work, a critical element to our approach is the existence of such a sensor. Therefore, we present a design of one such PV sensor. A worthwhile direction for future work is to analyze the characteristics (randomness and stability) of such sensors in greater detail.

In nano-scale transistors (especially for sub-65 nm technology nodes) the location and the number of dopants in the channel can vary widely between transistors (also known as random dopant fluctuations, or RDF [29]). RDF leads to different transistor threshold voltage (V_{th}) for transistors which are placed close to each other and designed to have the same threshold voltage. Measured properly using on-chip circuitry, such random variations in threshold voltage could act as a static random number. Conventionally, measurement of current difference between identical neighboring devices, followed by complex data analysis to extract V_{th} difference from the current difference, is used to characterize local random mismatch.

In this paper, we have used a sense-amplifier based test-circuit and measurement method for on-chip measurement of local random variation. Instead of complex and sophisticated analog voltage-current measurements required in conventional schemes, we used a simple digital measurement technique. The sensor is based on the current latch-type sense amplifier (CLSA) shown in Fig. 2(e) [30].

The circuit measures the difference in threshold voltage of the devices under test [circled in Fig. 2(e)] by applying a subthreshold reference voltage ($V_{\rm ref}$, generated on-chip) to the gate input of both devices. The sense amplifier, a cross-coupled inverter, determines the mismatch between the devices under consideration due to the exponential relationship of drain current on sub-threshold gate voltage and outputs a 0 or 1 based on the direction of the mismatch. Due to the use of small sized transistors for the DUTs, the mismatch is mainly due to RDF. The other transistors of the sensor are kept larger so that RDF has minimal effect on the sensor.

Simulations on a similar CLSA sensor using 130-nm technology [30] show that the two transistors could be differentiated by applying an offset voltage to the two devices to detect where an observable difference in current occurred. It was found that this offset voltage was strongly dependent on local random $V_{\rm th}$ variation while systematic (correlated) variation had minimal effect. While the offset voltage is a linear combination of the $V_{\rm th}$ offset of both the driver FETs ($N_{\rm DR} - N_{\rm DRB}$) and the mismatches in the latch FETs ($P_{\rm INV} - N_{\rm INV}, P_{\rm INVB} - N_{\rm INVB}$), the latch FETs are sized to be large such that RDF has minimal impact and does not lead to latch mismatches.

Since we would like the output to be stable under changes in temperature, let us now consider the stability of the random number generator to temperature variations.

To the first order, assuming no oxide charge, the threshold voltage of an nMOS transistor can be expressed as

$$V_{\rm th} = -\frac{E_g}{2q} + \psi_B + \frac{\sqrt{4\varepsilon_{si} q N_a \psi_B}}{C_{\rm ox}} \tag{1}$$

as derived by [31, p. 131]. E_g denotes the silicon bandgap, ψ_B is the surface-to-intrinsic potential, N_a is the channel doping density, and $C_{\rm ox}$ is the oxide capacitance. Both ψ_B and E_g are functions of temperature, although E_g is a very weak function of temperature. Substituting the expressions for ψ_B

$$\frac{\partial V_{\rm th}}{\partial T} = -(2m-1)\frac{k}{q}\left[ln\left(\frac{\sqrt{N_c N_v}}{N_a}\right)+\frac{3}{2}\right]+\left(\frac{m-1}{q}\right) \tag{2}$$

$$\frac{\partial E_g}{\partial T} \approx -(2m-1)\frac{k}{q}\left[ln\left(\frac{\sqrt{N_c N_v}}{N_a}\right)+\frac{3}{2}\right]. \tag{3}$$

Assuming that $\partial E_g/\partial T = 0$.

Typically, for $(N_c N_v)^{1/2} = 2.5 \cdot 10^{19}$ cm^{-3}, a channel doping $N_a = 1 \cdot 10^{16}$ cm^{-3} and a body coefficient $m = 1.1$, we have $\partial V_{\rm th}/\partial T = -1$ mV/K [31].

Now, let us consider the differential random number generator shown in Fig. 2(e). Let us consider that the threshold voltage difference due to the circuit under test is because of different number of discrete dopants in the channel leading to an average doping concentration of N_{a1} and N_{a2}. Let us consider a change in temperature ΔT (both transistors under test will experience the same temperature due to their close proximity). Under such a situation, if at temperature T, $V_{\rm th1} > V_{\rm th2}$, then we need to show for stability

$$V_{{\rm th1}@T+\Delta T} > V_{{\rm th2}@T+\Delta T}. \tag{4}$$

Mathematically modeling the temperature change of threshold voltage, the problem is equivalent to showing

$$V_{{\rm th1}@T+\Delta T} - V_{{\rm th2}@T+\Delta T} > \left(\frac{\partial V_{\rm th2}}{\partial T} - \frac{\partial V_{\rm th1}}{\partial T}\right) \cdot \Delta T \tag{5}$$

where $\partial V_{\rm th1}/\partial T$ and $\partial V_{\rm th2}/\partial T$ are estimated at T.

From the above analysis, since $\partial V_{\rm th}/\partial T \approx -1$ mV/K for similar N_a and m (N_{a1} and N_{a2} are almost similar, leading to a very similar m), the right-hand side of the above equation tends to zero. Experimental analysis [32] also shows differential thermal stability in the subthreshold region of operation.

While nMOS may behave differently under the effects of temperature as compared to pMOS, the temperature profiles of nearby transistors of the same type will be nearly identical, causing an equivalent temperature reaction on each side of the circuit as previously explained. This effect is likely to change the trip points of the two inverters equally, but will not cause appreciable differences in node capacitance since most capacitance in the circuit originates from the use of oxide in the MOSFET gates.

Altering the trip points of the transistors changes the total amount of charge the DUTs will need to drain before the cross-coupled inverters reinforce a voltage differential, but this effect is balanced between the two sides of the circuit since the capacitance at the internal nodes is unaffected by temperature. In addition, the cross-coupled inverters are of larger than minimum size, making them less susceptible to random dopant fluctuations. Hence, the design is quite insensitive to temperature fluctuations.

C. *Sensor Recovery for IC Unlocking*

The proposed design framework produces manufactured ICs that are inoperative, i.e., they will not produce correct functional outputs due to the effects of PV injection. In order to properly unlock and operate the fabricated chips, a unique unlocking key must be computed based on the PV sensor outputs. While the easiest method for exposing the values generated by PV sensors is to include their digitized outputs in a DFT structure such as a scan chain, such a path could be used just as easily by attackers as by authorized users. Therefore, we instead generate and use a set of test vectors specifically constructed to expose the effect of the PV sensors at the primary outputs (or scanned state elements).

The unlocking procedure is detailed in Fig. 5. The test database distributes both a universal set of primary input vectors and a generic set of internal key vectors. Prior sensor readings are combined with the generic internal keys vectors to create the true internal key vector for testing on the modified logic core. The internal key is not directly applied due to the preprocessor; instead, the equivalent unlocking key must be created, which is accomplished by performing the inverse operation of the preprocessor. The response bits are accumulated and then matched

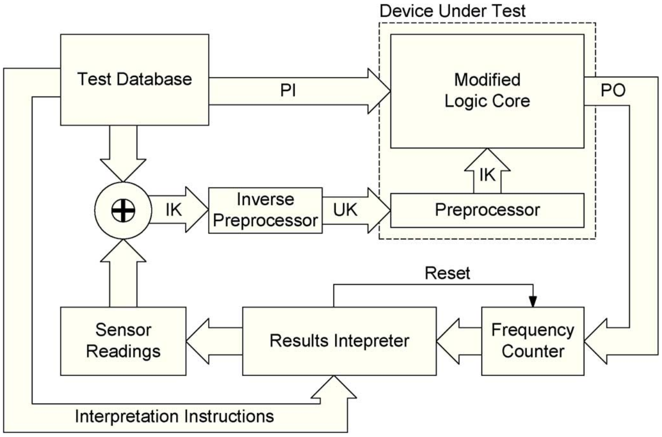


Fig. 5. Unlocking procedure. Test vectors are generated dynamically by merging predetermined test vectors with known sensor readings and computing the required unlocking key to supply to the DUT. The outputs undergo a frequency analysis that reveals additional sensor values.

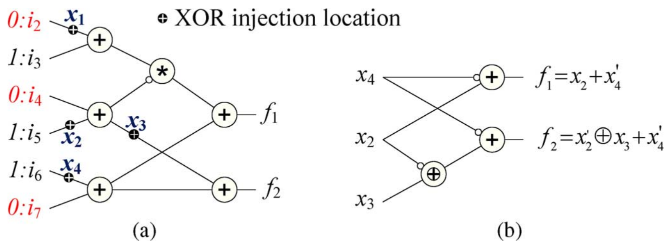


Fig. 6. Random logic pruning on binary logarithm circuitry. (a) Pre-pruned logic with marked injection locations. (b) Post-pruned logic in terms of only injection values. If the gates are sorted topologically, we can propagate constant values from the primary inputs through the gates, eliminating gates that receive constant inputs at a linear rate.

against a table to determine the value of sensors in the chip. These sensor values are then used as part of subsequent tests to obtain more test vectors. Once the procedure is complete, the chip-specific unlocking key can be calculated by supplying one additional generic internal key from the test database, which when combined with the internal key and fed through the inverse processor, produces the chip-specific unlocking key.

The formulation of the test vectors cannot use standard test tools—it may be impossible to uniquely expose a sensor's value at the output. The test generator must assume that the non-determined sensors are represented as an unknown state, and generate a pattern at the primary input that creates a controlling path from the sensor to the output, while simultaneously ensuring the sensor modifies a fixed (variation-free) value at the switchbox. Such a test, potentially implemented via stuck-at fault testing tools, would yield an output which would be in a direct relationship with the PV sensor's value. However, it is quite likely that not all sensors can uniquely sensitize an output, and as such, SAT techniques are not adequate.

Since the node selection heuristics are likely to be geared towards maximal functional impact, random vector simulation at the primary inputs is likely to expose the sensor's values at the outputs. Rather than assuming the sensors are an unknown value, the vectors instead propagate through the logic, pruning the logic by eliminating any gates with constant inputs such that the logic becomes a function of only a composite vector $(X = V \oplus K)$ between the PV sensor readings V and the internal key K.[2] Such pruning will result in each bit of the primary output $(f_i \in F)$ becoming a function of specific bits of the composite vector $(\{x\} \in X)$. If f_i is a function of only one composite x_j, the given test pattern is effective at finding the sensor value v_j.

In practice, not all sensors find such a trivial route to the output, but instead, are only visible in multivariable output functions. Determining the values might be possible through linear programming, or alternatively, through exhaustive key toggling. If we assume one of the inputs (e.g., x_i) is fixed at either a logical 0 or 1, we can find the expected value of the function $E(f)$ under each scenario by enumeration of the set S of potential inputs to the other inputs $(X \setminus x_i)$.

If a mismatch exists between $E(f_{x_i'})$ and $E(f_{x_i})$ for some i, it is possible to determine v_i from the function. Even without knowledge of x_i's value, fixing the key input k_i and applying the set of test vectors S will yield an output probability equal to one of the expected values, revealing what x_i (and, consequently, v_i) is. While we cannot directly apply distinct values in the set S to the composite vector X since we do not yet know V, applying the set to K will still apply the entire set at X, but will permute the set due to the variations V. Once v_i is found, the function can be partitioned again to find a different expected value mismatch with a set half its previous size.

After repeating the mismatch characterization on other outputs, the pruning process is repeated, but with an advantage: for each sensor v_i we could determine on prior passes, the composite value x_i can be deterministically set to allow for deeper circuit pruning: we simply need to apply $x_i \oplus v_i$ to k_i, and x_i arrives at the desired value.

Consider the circuit in Fig. 6(a) which is part of a binary logarithm circuit. By propagating the shown random input vector through the logic, we arrive at the pruned circuit shown in Fig. 6(b) that is in terms of only the input combinations. For f_2, partitioning by both x_2 and x_3 causes no difference between the expected values (see Table II); however, partitioning by x_4 does result in a noticeable outcome. For f_1, more information can be gathered—partitioning by x_2 immediately reveals a distinction in expected values, and a further partition reveals x_4 independently of analyzing f_2. With the knowledge of x_2, x_3 could then be determined from f_2. While x_1 is not observable in the output, it could be determined through a separate pruning pass.

Our testing method's downside is twofold: an output of n distinct sensors that can all be determined requires up to $2^{n-1} + 2^{n-2} + \cdots = 2^n - 1$ different keys for a single primary input vector to determine the sensor values, and due to the preprocessor and the dynamic nature of the test procedure, the test vectors applied to the chip must be partially constructed during the test phase. However, this exponential growth is localized, and can be compensated for by simply ignoring any outputs that exceed a predetermined input threshold. The test

[2] Simplifying the sensor and the key as a single variable is only possible with XOR-based switchboxes.

TABLE II
EXPECTED VALUES ON PRUNED CIRCUIT

pattern	$\mathrm{E}(f_1)$	pattern	$\mathrm{E}(f_1)$
x'_2	0.5	x_2	1.0
$x'_2 x'_4$	1.0	$x'_2 x_4$	0.0

pattern	$\mathrm{E}(f_2)$	pattern	$\mathrm{E}(f_2)$
x'_2	0.75	x_2	0.75
x'_3	0.75	x_3	0.75
x'_4	1.0	x_4	0.5
$x'_2 x_4$	0.5	$x'_2 x_4$	0.5
$x'_3 x_4$	0.5	$x'_3 x_4$	0.5
$x'_2 x'_3 x_4$	1.0	$x'_2 x_3 x_4$	0.0

set can be further reduced by selecting optimal test sets that maximize the number of sensor values revealed per test vector.

D. Key Preprocessor

The trivial relationship between the variation sensors and the key is a problematic vulnerability. With circuit-level knowledge and an understanding of the original, intended functionality, an attacker could use the testing process described in Section III-C just as easily as the IP rights owner. Even without circuit-level knowledge but a functional understanding, an attacker could perform annealing on the key bits with respect to the output error. However, if one can control the application of keys, such a vulnerability can be overcome.

The *internal keys* that unlock a chip's functionality are little more than messages that require delivery to a location inside the chip. When applying an external, *unlocking key* to the system, the unlocking information traverses a key preprocessor prior to application to the circuit. This preprocessor can be implemented in two methods: either through decryption or through authentication of the unlocking key.

Key decryption protects key application by encrypting the internal key off-chip, and supplying the encrypted information as the unlocking key. The preprocessor would then decrypt the unlocking key, and apply the decrypted internal key to the logic.

Selection of a decryption preprocessor depends heavily on what can be afforded by the designer: asymmetric implementations occupy far more area than symmetric implementation (e.g., 10 k gates for ECC versus 3.4 k gates for AES [33]), but due to the fixed implementation in silicon, the preprocessor's decryption key will be hardcoded. Under a symmetric implementation, knowledge of this decryption key implies immediate knowledge of the encryption key, and consequently, the transform from internal to unlocking key. With an asymmetric implementation, knowledge of the decryption key conveys no information about the inverse transformation required to construct the unlocking key.

Key authentication involves supplying a signature alongside the internal key. The circuitry could forbid the application of a key without a proper signature, or it could feed the results of the signature check directly into the logic, inserting primitive gates elements to block or modify logic values as they traverse the logic. If the primitive gates are placed in certain paths, they could prevent the exposure of the internal key by not allowing the sensors' values to propagate to the outputs, and therefore, make the internal key undiscoverable. Alternatively, different signature parameters could allow for a higher density of logic modifications by segmenting pruned circuits during the testing phase.

IV. Experimental Results

The proposed methodology was evaluated by implementing the CLIP design flow described in this paper within the ABC [34] logic synthesis framework. Given a gate-level circuit, the identification and insertion of injection and correction nodes are automatically performed. Next, the test vectors used for key recovery are automatically generated. We applied the methodology to several ISCAS'89 and ITC'99 benchmark circuits. Three different node selection heuristics were used in our analysis: maximal functional impact, heavy grouping, and converse power. Maximal impact chose nodes based on greedily maximizing the hamming distance between the output when an incorrect key is applied and the correct output. Heavy grouping concentrated on choosing nodes such that the effects of sensors have greater inter-dependencies (and are harder to distinguish). The converse power heuristic chose nodes such that the difference in switching activity of an incorrect key versus a correct key is minimal or possibly negative.

There are four distinct areas of interest with respect to the impact our methodology has on the circuit: functional impact (how much the outputs differ from the correct values when an incorrect key is applied), area, delay, and power. Since our heuristics were designed to avoid affecting the critical paths of the circuit, the delay of the resultant circuitry was unchanged in all our experiments.[3] We present an evaluation of the other metrics.

Our results are divided into two parts—area results are presented for several benchmarks, while extensive analysis is performed for one of the benchmarks in order to present deeper insights. To simplify the extensive analysis, the s5378 benchmark circuit was selected for demonstration purposes due to its manageable size (988 gates), and an area overhead limit of 20% was used when inserting injection and correction nodes.

A. Area Impact

Table III presents the area overheads of CLIP for a set of benchmark circuits to achieve various key lengths. For the results presented in this table, the maximal impact heuristic was used for selection of injection and correction nodes, along with a strict constraint that the delay of the circuit should not increase (i.e., nodes selected for injection and correction should have sufficient slack). We present area overheads both without and with the PV sensors, calculated based on the assumption that each sensor is equivalent to 3.5 logic gates [see Fig. 2(e)]. In the case of the x3 benchmark, our tool was unable to realize a 256-bit key since there was not enough delay slack to support sufficient injection and correction nodes.

[3] For some circuits, it is possible that the critical path(s) may have to be impacted to insert a sufficient number of injection/correction nodes and achieve a sufficiently large key size. In such cases, the designer needs to make the tradeoff between delay and security.

TABLE III
AREA OVERHEAD FOR SPECIFIC KEY STRENGTHS

		Node Usage			with sensor estimate		
Circuit	Nodes	16 bits	64 bits	256 bits	16 bits	64 bits	256 bits
x3	634	687	811	N/A	743	1035	N/A
dalu	1103	1188	1448	2054	1244	1672	2950
C5315	1310	1383	1606	2252	1439	1830	3140
des	3571	3628	3792	4320	3684	4016	5216
s38417	7892	7995	8216	9007	8051	8440	9903

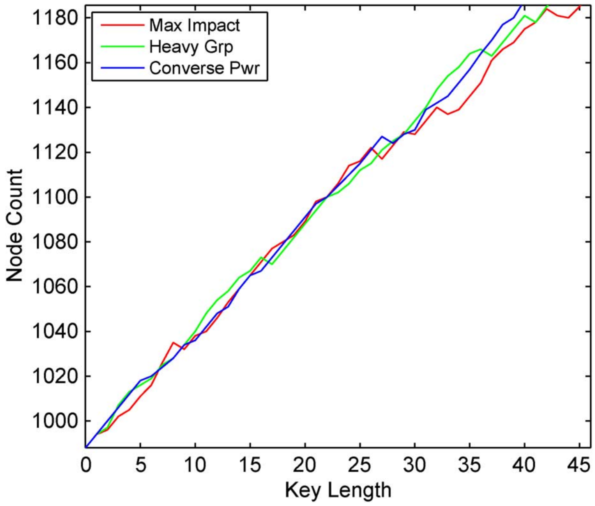


Fig. 7. Area overhead for various heuristics on the benchmark circuit S5378 with a 20% area limit for logic modification.

There are three parts to the area overhead incurred by the proposed method: logic modification, PV sensors, and the key preprocessor, of which we have quantified the first two factors in Table III. Since the preprocessor that a designer would use is independent of our methodology, and we anticipate that it would be shared across several blocks in an IC, we did not include it in our overhead calculations. However, for completeness we note that compact implementations of symmetric and asymmetric encryption algorithms have been proposed (e.g., 10 k gates for ECC and 3.4 k gates for AES [33]), which should not present a major overhead in medium to large ICs. If the IC already includes a hardware or software based encryption block as part of its functionality, we could simply reuse the same as the preprocessor (at powrup) without incurring any overheads.

Fig. 7 shows the impact of the different selection heuristics on area overheads for the s5378 benchmark circuit, and suggests that all of the heuristics lead to reasonable increases in circuit area.

B. Functional Impact

The functional impact of our methodology can be observed in Fig. 8. With only a small number of incorrect key bits, the output achieves incorrect operation at all times. The average Hamming distance can be seen in Fig. 9, as well as a *wrapped* hamming distance, where nodes with greater than 50% distance count as 100%-*distance*. Since the maximal impact heuristic did not achieve a higher impact than other heuristics, such a greedy selection algorithm is unsuitable.

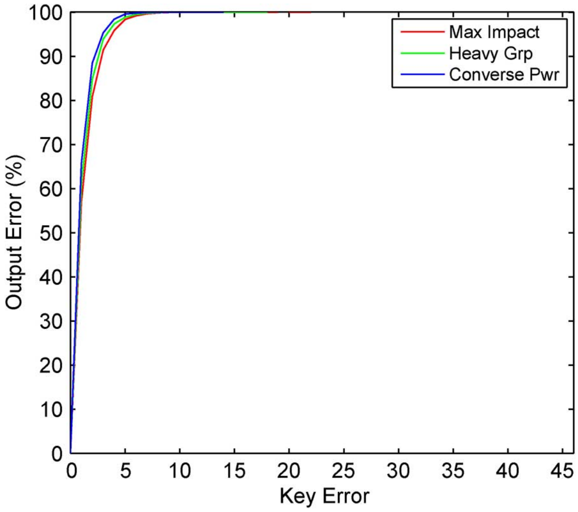


Fig. 8. Functional impact. Only a small number of wrong key bits ensures the circuitry does not give proper results.

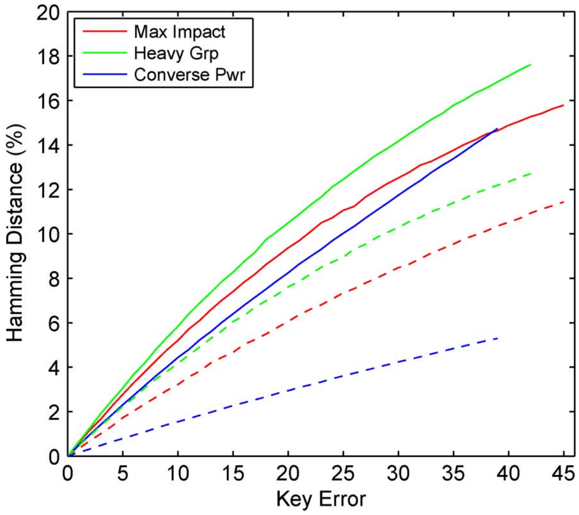


Fig. 9. Output Hamming distance based on key error. Dashed results represent a wrapped Hamming distance measure.

C. Power Impact

The impact of CLIP on power consumption (quantified by the additional switching activity in the CLIP enhanced circuit as compared to the original circuit) is presented in Fig. 10 for the s5378 benchmark. In general, the switching overhead rises at a rate on par with the additional logic required. As discussed in the next section, the additional power consumed varies with the number of bits that are incorrect in the internal key, leading to the possibility of differential power analysis attacks unless a preprocessor is used.

D. Key Recovery

To expose the sensor values, different vector combinations were applied in an attempt to maximize the number of bits recovered for each test vector applied. It was observed that with the S5378 benchmark, only 6–8 test vectors were necessary to recover all 45 sensor values. Each test vector used for sensor

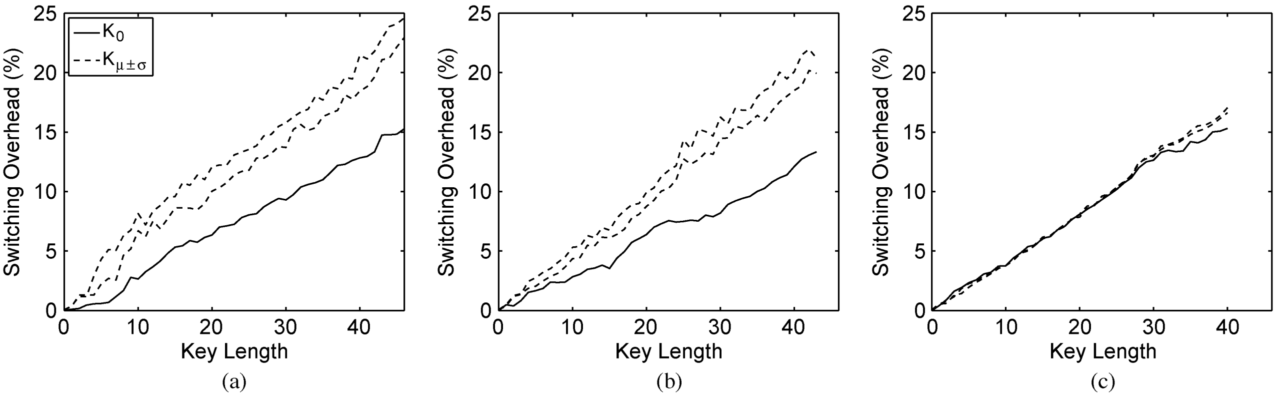


Fig. 10. Switching overhead for increasing key length. (a) Maximal impact. (b) Heavy grouping. (c) Converse power.

extraction was obtained through a guided simulation based procedure, choosing test vectors that propagate the largest number of PV sensor values to the circuit outputs at a time.

V. Security Analysis

In order to understand the benefits of the CLIP framework, we analyze it against several different styles of attacks to determine its effectiveness. A number of attack models were considered, including external attacks such as brute force, man in the middle, differential power analysis, and differential signal analysis, and internal attacks such as mask modification based on varying levels of knowledge.

A. External Attacks

External attacks view the chip as a black box, i.e., the attacker has no knowledge of design internals, and the cost of de-packaging the chip and reverse engineering the design are too high. The simplest external attack is the *brute force* attack—if an attacker only has external access to a locked chip (and knows the expected output for a given input), they may attempt to discover the unlocking key by enumerating all possible values until they find a combination that causes the locked chip to reach an unlocked state (i.e., behave like an unlocked chip). Due to the low overhead associated with increasing the key length, a designer is likely to choose key lengths that exceed reasonable levels of computing power. Therefore, brute force attacks are infeasible and an attacker must seek out more intelligent approaches.

Differential signal analysis attacks attempt to apply multiple inputs and keys that differ in a controlled manner (e.g., at specific bit positions), and analyze the differences between the chip responses to them. They have been used successfully to reduce the complexity of breaking cryptographic algorithms [35]. In our context, differential analysis is a concern if there are simple interdependencies between the unlocking key bits and the IC outputs (e.g., if one or a few of the bits of the unlocking key affect only one or a few output bits). Such attacks are precisely the reason for our use of a cryptographic preprocessor to de-couple the unlocking key from the internal key. We illustrate this by performing experiments on several benchmarks where we removed the preprocessor, and attempted to apply simulated annealing to search for the key value that minimizes the error in the output bits (difference between the incorrect outputs and the expected correct output values). This approach was successful in unlocking the protection in several benchmarks when no preprocessor was used. Fig. 11(a) shows an example of multiple simulated annealing attempts on the s5378 benchmark. However, the addition of a preprocessor makes it extremely difficult to perform such attacks. Fig. 11(b) demonstrates annealing attempts on the same benchmark, using a very simple preprocessor based on a random network of XOR gates in a pseudo-decryption configuration. While we recommend the use of preprocessors based on cryptographic algorithms in practice, our experiment shows that even a trivial preprocessor is effective in defending against differential analysis attacks.

Man in the middle and *replay* attacks rely on intercepting the stream of information passed to the chip with hopes of either modifying the bitstream or reusing the information transmitted. We consider man in the middle attacks launched by observing the operation of an unlocked chip, as well as attacks launched by observing the key recovery process. If the unlocking key is stored off-chip and transferred to the chip upon powerup, the unlocking key could be intercepted, but this would be of no value since the key is valid only for the given chip.

Man in the middle attacks could also be applied by an attacker who has observed the test vectors applied to a chip during the key recovery process. The same test vectors could be applied to any chip, and will result in the propagation of many of the PV sensor values to the circuit outputs. However, without a knowledge of the design (gate-level netlist and exact location of injection and correction nodes), this information is of little use. Furthermore, even if the attacker is able to infer the values of PV sensors from the circuit responses, the attacker must apply the inverse of the preprocessor function in order to convert these values into an unlocking key, which is equivalent to breaking the cryptographic algorithm used in the preprocessor.

Differential power analysis (DPA) [36] analyzes a circuit's power trace in order to expose information about the key. In the context of CLIP, DPA could result in the exposure of the preprocessor's decryption key. DPA-resistant design technques have been extensively researched in recent years, and many implementations have been proposed [37], [38] that could be applied to the implementation of the preprocessor to address such an attack. In addition, the injection and correction nodes in the

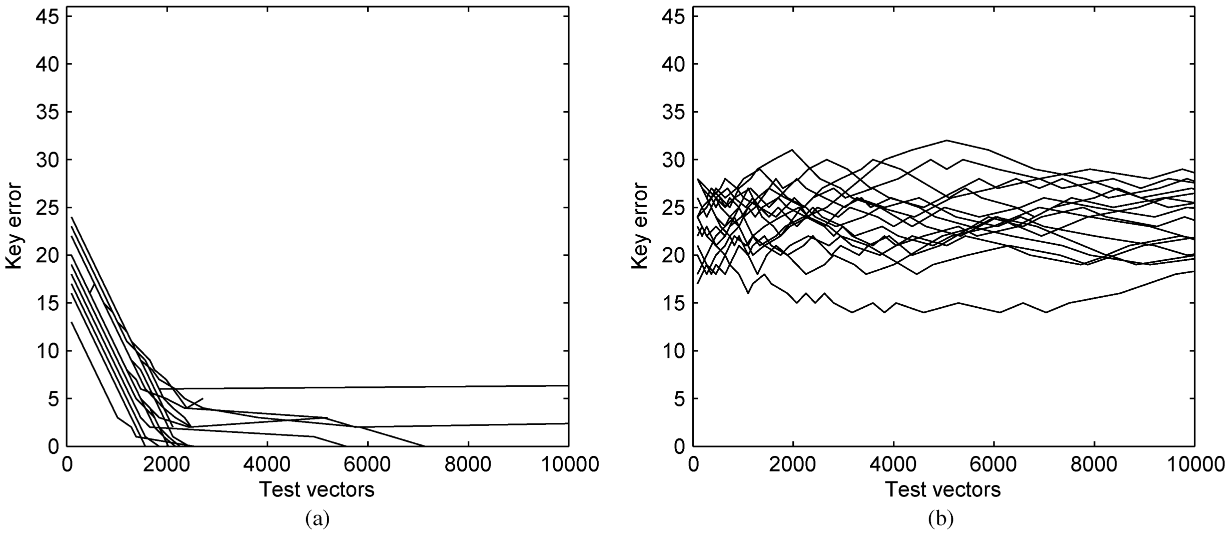


Fig. 11. Annealing attacks on s5378. (a) No preprocessor. (b) XOR-based preprocessor. The preprocessor used approximately 130 XOR gates (25% additional area) to dissociate the unlocking and internal keys.

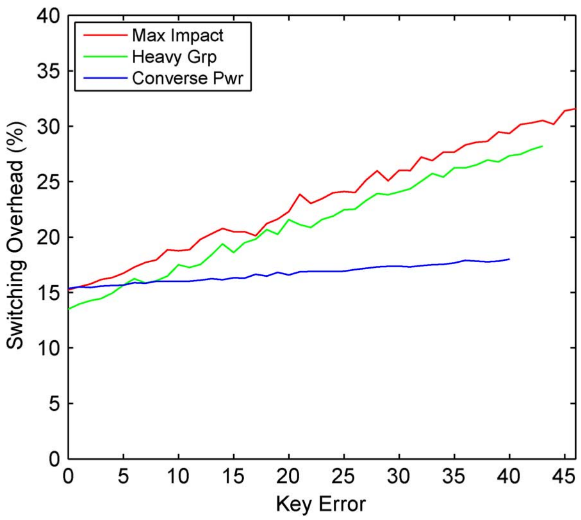


Fig. 12. Switching overhead based on key error.

circuit are also vulnerable to power analysis; unbiased node selection reveals a linear relationship between the average power usage and the number of correct bits (see Fig. 12). However, we note that exploiting this correlation will require that the attacker apply *internal* keys with specific characteristics, which is prevented by the use of the cryptographic preprocessor.

B. Mask Knowledge and Internal Attacks

External attacks all have one major weakness: limited reusability. There is low value in discovering the unlocking key for only one chip; each chip has distinct intra-die variations that cause unique sensor measurements, and consequently, a unique unlocking key. Even a successful external attack would require that the attacker incur significant per-chip costs.

Internal attacks require a higher up-front cost (such as chip reverse-engineering and mask extraction), or the collusion of an entity that has a knowledge of the mask (the foundry, or one of its employees). However, they are attractive from an attacker's perspective since they could lead to elimination of per-chip costs to unlock chips.

We consider microprobing attacks, mask analysis and modification attacks, and combinations thereof, and discuss the difficulty of launching each on an IC protected by CLIP.

Microprobing refers to depackaging of a chip followed by probing of internal signals (e.g., using e-beam microscopy) while the chip is operational. Advanced microprobing attacks may also make small modifications to the circuit (e.g., by cutting or shorting wires). First, we note that there is no constant secret value that can be revealed by probing any chip protected by CLIP, since all secrets differ from chip to chip. This implies that probing a chip can only lead to unlocking that single chip itself, which is not a viable proposition given the cost of the equipment involved. Furthermore, any process used to read out the values of the PV sensors should be sensitive enough that it does not disturb the relationship between the adjacent transistors in the sensor, and this process must be repeated at several spots in the chip since the sensors are dispersed throughout the logic.

Mask analysis and modification attacks involve studying the mask to extract information that can be used in an external or internal attack, or modifying the mask to disable or significantly weaken the protection scheme. For unsophisticated attackers, the small size of the PV sensors coupled with their spread throughout the logic makes mask analysis difficult. If the locations of the PV sensors are pinpointed by sophisticated mask analysis, the attacker still needs to undo all changes made to the circuit during the CLIP design process—merely removing the sensors and hardwiring their outputs to fixed values will not help, since the correction nodes will now effectively ensure that the circuit is still not operational. Discovering the location of correction nodes is difficult since there is no fixed pattern (gate type) that the attacker can look for, and they can be placed at arbitrary distance from the PV sensors. Even if the correction nodes are located, the relationship between the PV sensor outputs and internal key (determined during the synthesis process) is not trivial, and must be discovered.

If one targets the preprocessor instead of the sensors, this is of little utility. Unless the designer selects a symmetric encryption algorithm for the preprocessor, mask analysis of the preprocessor to expose the decryption key does not help—revealing the decryption key in an asymmetric algorithm does not provide adequate information for the inverse transformation required to construct the unlocking key, and as such, the system is still secure. If the decryption key is recovered for a symmetric algorithm, this key may be as useful to an attacker as bypassing the preprocessor altogether, but regardless of recovering a symmetric key or remanufacturing a chip to bypass the preprocessor, each manufactured chip still requires extensive post-manufacturing characterization for constructing an unlocking key since the sensors themselves were not removed.

It must be noted that the objective of most security schemes in practice is to raise the difficulty and cost of launching an attack to a point where the attack is no longer attractive, while keeping the cost of adopting the security scheme minimal. We now describe a couple of sophisticated attacks that can be used to break CLIP, which we believe are of sufficiently high cost and difficulty.

In order to have complete reusability for an attack, an attacker could purchase an unlocked chip (thereby legitimately acquiring an unlocking key for it), successfully read out the PV sensor values, extract the mask, refabricate the chip with PV sensor outputs replaced by the fixed values, and utilize the same unlocking key for all chips. Such an attack requires that the aforementioned issue of nondestructive reads of PV sensors be overcome. It is very difficult to protect against such an elaborate attack.

Another possible attack is to analyze the mask to locate the PV sensors, replace them with fixed values, and feed a chip through the standard key discovery process. We note that such an attack will require the participation of the owner of the IP rights (since the responses of the chip during the key discovery process need to be converted to an unlocking key), and can be prevented by ensuring that the key discovery and test process are secured using techniques similar to [13] and [26]. At the very least, a record of unlocking keys that have been issued can be used to help establish liability a posteriori.

In summary, we believe that CLIP substantially raises the cost and difficulty of IC piracy. In addition, as discussed in Section II, CLIP offers significant advantages over other active protection schemes due to several factors such as preserving the uniqueness of the applied keys at all levels, the distributed nature of PV injection and correction, and the use of a cryptographic preprocessor to decouple the internal and external unlocking keys.

VI. Summary

In this paper we introduced a methodology that provides a low-overhead solution to IC locking while simultaneously requiring a unique, per-chip unlocking key derived from embedded PV sensor readings. The output of each sensor is fed directly into a block of random logic to alter the logic's operation, and with the help of logic modification and additions (localized duplication, switchboxes, and selectors), we added a means to correct for the altered operation caused by the process variations. When providing an unlocking key to the design, the unlocking key is transformed by means of preprocessor circuitry into an internal key, which, if correct, allows the circuitry to function as originally intended.

Our methodology offers strong device locking at the circuit level and ensures a unique internal key for each chip. Simulation results showed that the methodology achieves scalable security, and the small overhead associated with implementing CLIP in the example circuits shows that it is quite suitable for adoption in practice.

References

[1] V. Alliance, "Intellectual property protection: Schemes, alternatives, and discussion," Aug. 2000.

[2] *Protection of Semiconductor Chip Products.* US Code, Title 17, Chapter 9.

[3] *The Act Concerning the Circuit Layout of a Semiconductor Integrated Circuit*. Japan, Act No. 43, 1985.

[4] *The Legal Protection of Topographies of Semiconductor Products.* European Union, European Council Directive 87/54/EEC..

[5] *Agreement on Trade-Related aspects of Intellectual Property Rights*. Marrakesh Agreement Establishing the World Trade Organization, Annex 1C.

[6] R. Torrance and D. James, "The state-of-the-art in IC reverse engineering," in *Proc. CHES*, 2009, pp. 363–381.

[7] E. Charbon, "Hierarchical watermarking in IC design," in *Proc. IEEE Custom Integr. Circuit Conf.*, 1998, pp. 295–298.

[8] A. Kahng, J. Lach, W. H. Mangione-Smith, S. Mantik, I. L. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe, "Watermarking techniques for intellectual property protection," in *Proc. DAC*, 1998, pp. 776–781.

[9] A. Oliveira, "Robust techniques for watermarking sequential circuit designs," in *Proc. DAC*, 1999, pp. 837–842.

[10] Y. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security," in *Proc. USENIX Security Symp.*, 2007, pp. 1–16.

[11] Y. Alkabani, F. Koushanfar, and M. Potkonjak, "Remote activation of ICs for piracy prevention and digital right management," in *Proc. ICCAD*, 2007, pp. 674–677.

[12] R. Chakraborty and S. Bhunia, "Hardware protection and authentication through netlist level obfuscation," in *Proc. ICCAD*, 2008, pp. 674–677.

[13] J. Roy, F. Koushanfar, and I. Markov, "EPIC: Ending piracy of integrated circuits," in *Proc. DATE*, 2008, pp. 1069–1074.

[14] J. Huang and J. Lach, "IC activation and user authentication for security-sensitive systems," in *Proc. HOST*, 2008, pp. 76–80.

[15] O. Sinanoglu and A. Orailoglu, "Partial core encryption for performance-efficient test of SOCs," in *Proc. ICCAD*, 2003, pp. 91–94.

[16] C. Paar, J. Guajardo, S. Kumar, and T. Guneysu, "Secure IP-block distribution for hardware devices," in *Proc. HOST*, 2009, pp. 82–89.

[17] A. T. Abdel-Hamid, S. Tahar, and E. M. Aboulhamid, "A public-key watermarking technique for IP designs," in *Proc. DATE*, 2005, pp. 330–335.

[18] P. S. Ravikanth, "Physical one-way functions," Ph.D. dissertation, Media Art Sci., Massachusetts Inst. Technol., Cambridge, 2001.

[19] G. E. Suh, C. W. O'Donnel, I. Sachdev, and S. Devadas, "Design and implementation of the AEGIS single-chip secure processor using physical random functions," in *Proc. ISCA*, 2005, pp. 25–36.

[20] S. S. Kumar, J. Guajardo, R. Maes, G. Schrijen, and P. Tuyls, "The butterfly PUF protecting IP on every FPGA," in *Proc. HOST*, 2008, pp. 67–70.

[21] J. Guajardo, S. S. Kumar, G. Schrijen, and P. Tuyls, "FPGA intrinsic PUFs and their use for IP protection," in *Proc. CHES*, 2007, pp. 63–80.

[22] B. Gassend, D. Clark, M. van Dijk, and S. Devadas, "Delay-based circuit authentication and applications," in *Proc. ACM Symp. Appl. Comput.*, 2003, pp. 294–301.

[23] Y. Su, J. Holleman, and B. Otis, "A digital 1.6 pJ/bit chip identification circuit using process variations," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, pp. 69–77, Jan. 2008.

[24] D. Roy, J. H. Klootwijk, N. A. M. Verhaegh, H. H. A. J. Roosen, and R. A. M. Wolters, "Comb capacitor structures for on-chip physical uncloneable function," *IEEE Trans. Semicond. Manuf.*, vol. 22, no. 1, pp. 96–102, Feb. 2009.

[25] F. Koushanfar and G. Qu, "Hardware metering," in *Proc. DAC*, 2001, pp. 490–493.

[26] R. Maes, D. Schellekens, P. Tuyls, and I. Verbauwhede, "Analysis and design of active IC metering schemes," in *Proc. HOST*, 2009, pp. 74–81.

[27] Certicom Corp., Mississauga, ON, Canada, "Certicom launches trusted key injection platform for anti-cloning," 2005. [Online]. Available: http://www.certicom.com/

[28] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," in *Proc. CRYPTO*, 1991, pp. 2–21.

[29] D. Reid, C. Millar, G. Roy, S. Roy, and A. Asenov, "Analysis of threshold voltage distribution due to random dopants: A 100,000-sample 3-D simulation study," *IEEE Trans. Electron Devices*, vol. 56, no. 10, pp. 2255–2263, Oct. 2009.

[30] S. Mukhopadhyay, K. Kim, K. Jenkins, C. Chuang, and K. Roy, "Statistical characterization and on-chip measurement methods for local random variability of a process using sense-amplifier-based test structure," in *Proc. ISSC*, 2007, pp. 400–611.

[31] Y. Taur and T. H. Ning, *Fundamentals of Modern VLSI Devices*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[32] P. Andricciola and H. Tuinhout, "The temperature dependence of mismatch in deep-submicrometer bulk MOSFETs," *IEEE Electron Device Lett.*, vol. 30, no. 6, pp. 690–692, Jun. 2009.

[33] T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann, and L. Uhsadel, "A survey of lightweight-cryptography implementations," *IEEE Des. Test*, vol. 24, no. 6, pp. 522–533, Jun. 2007.

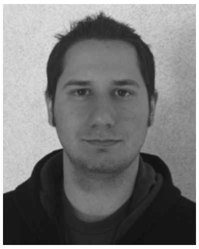
[34] Berkeley Logic Synthesis and Verification Group, Berkeley, CA, "ABC: A system for sequential synthesis and verification, Release 70930," 2007. [Online]. Available: http://www.eecs.berkeley.edu/~alanmi/abc/

[35] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," in *Proc. CRYPTO*, 1991, pp. 2–21.

[36] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proc. CRYPTO*, 1999, pp. 388–397.

[37] H. Saputra, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, R. Brooks, S. Kim, and W. Zhang, "Masking the energy behavior of DES encryption," in *Proc. DATE*, 2003, pp. 84–89.

[38] K. Tiri and I. Verbauwhede, "A logic level design methodology for a secure DPA resistant ASIC or FPGA implementation," in *Proc. DATE*, 2004, pp. 246–251.



W. Paul Griffin received the B.S. degree in mathematics from Evangel University, Springfield, MO, in 2007, and is currently pursuing the Ph.D. degree from the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN.

In 2011, he was a research intern with the Security Center of Excellence, Intel, Hillsboro, OR. His research interests vary from logic to system-on-chip designs, with influence towards security applications.

Anand Raghunathan (F'10) received the B.Tech. degree in electrical and electronics engineering from the Indian Institute of Technology, Madras, India, in 1992, and the M.A. and Ph.D. degrees in electrical engineering from Princeton University, Princeton, NJ, in 1994 and 1997, respectively.

He is currently a Professor with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN. Previously, he was a Senior Research Staff Member with NEC Laboratories America, Princeton, NJ, where he led research projects related to system-on-chip architectures, design methodologies, and design tools. He has coauthored the book *High-level Power Analysis and Optimization* (Springer, 1997) and six book chapters, and has presented several full-day and embedded conference tutorials in the above areas. He holds 20 U.S. patents in the areas of advanced system-on-chip architectures, design methodologies, and VLSI CAD.

Dr. Raghunathan was a recipient of seven Best Paper Awards and three Best Paper Nominations at leading conferences. He received a Patent of the Year Award (an award recognizing the invention that has achieved the highest impact) and a Technology Commercialization Award from NEC in 2001 and 2005, respectively. He was a recipient of the IEEE Meritorious Service Award (2001) and Outstanding Service Award (2004). He was chosen by MIT's Technology Review among the TR35 (top 35 innovators under 35 years, across various disciplines of science and technology) in 2006, for his work on "making mobile secure". He has been a member of the technical program and organizing committees of several leading conferences and workshops. He has served as Program and General Co-chair for the ACM/IEEE International Symposium on Low Power Electronics and Design, the IEEE VLSI Test Symposium, and the IEEE International Conference on VLSI Design. He has served as Associate Editor of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, *ACM Transactions on Design Automation of Electronic Systems*, IEEE TRANSACTIONS ON MOBILE COMPUTING, *ACM Transactions on Embedded Computing Systems*, IEEE DESIGN AND TEST OF COMPUTERS, and the *Journal of Low Power Electronics*. He is a Golden Core Member of the IEEE Computer Society.

Kaushik Roy (F'01) received the B.Tech. degree in electronics and electrical communications engineering from the Indian Institute of Technology, Kharagpur, India, and the Ph.D. degree from the Electrical and Computer Engineering Department, University of Illinois at Urbana-Champaign, in 1990.

He was with the Semiconductor Process and Design Center, Texas Instruments, Dallas, where he worked on FPGA architecture development and low-power circuit design. He joined the Electrical and Computer Engineering Faculty, Purdue University, West Lafayette, IN, in 1993, where he is currently a Professor and holds the Roscoe H. George Chair of Electrical and Computer Engineering. His research interests include Spintronics, VLSI design/CAD for nano-scale Silicon and non-Silicon technologies, low-power electronics for portable computing and wireless communications, VLSI testing and verification, and reconfigurable computing. Dr. Roy has published more than 500 papers in refereed journals and conferences, holds 15 patents, graduated 50 Ph.D. students, and is co-author of two books on low power CMOS VLSI design.

Dr. Roy was a recipient of the National Science Foundation Career Development Award in 1995, IBM Faculty Partnership Award, ATT/Lucent Foundation Award, 2005 SRC Technical Excellence Award, SRC Inventors Award, Purdue College of Engineering Research Excellence Award, Humboldt Research Award in 2010, and Best Paper Awards at 1997 International Test Conference, IEEE 2000 International Symposium on Quality of IC Design, 2003 IEEE Latin American Test Workshop, 2003 IEEE Nano, 2004 IEEE International Conference on Computer Design, 2006 IEEE/ACM International Symposium on Low Power Electronics and Design, and 2005 IEEE Circuits and System Society Outstanding Young Author Award (Chris Kim), 2006 IEEE Transactions on VLSI Systems Best Paper Award. He is a Purdue University Faculty Scholar. He was a Research Visionary Board Member of Motorola Labs (2002) and held the M.K. Gandhi Distinguished Visiting faculty at Indian Institute of Technology (Bombay). He has been on the editorial board of IEEE DESIGN AND TEST, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, and IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS. He was Guest Editor for the Special Issue on Low-Power VLSI in the IEEE Design and Test (1994) and IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS (June 2000), IEEE PROCEEDINGS—COMPUTERS AND DIGITAL TECHNIQUES (July 2002).