

Preventing IC Piracy Using Reconfigurable Logic Barriers

Alex Baumgarten
Microsoft

Akhilesh Tyagi and Joseph Zambreno
Iowa State University

Editor's note:

Hardware metering to prevent IC piracy is a challenging and important problem. The authors propose a combinational locking scheme based on intelligent placement of the barriers throughout the design in which the objective is to maximize the effectiveness of the barriers and to minimize the overhead.

—*Farinaz Koushanfar, Rice University*

Texas Instruments and AMD that once fabricated their own ICs have gone fabless, and other fabless companies such as Qualcomm, Broadcom, and Nvidia have broken into the top 20 semiconductor sales leaders. This paradigm shift has created a serious security threat

■ **A SIGNIFICANT SECURITY THREAT** has emerged because of changes to chip fabrication introduced from the flattening of the once-vertical IC supply chain. In the past, IC design and fabrication were typically handled by the same entity, because the cost to build a foundry, though expensive, was a reasonable investment. However, decreasing feature size and time to market, coupled with demands for lower-power, high-performance ICs, have made the cost required to establish a full-scale foundry prohibitive. In 2005, a full-scale 300-mm-wafer, 65-nm-process foundry cost \$3 billion to build. Very few companies can afford such an expense, thus driving the market to specialize. IP vendors have emerged that specialize in creating functional units that they license to IC designers for their ASIC designs. IC design companies integrate third-party IP along with their own to create an IC design. Finally, contract foundries harness economies of scale, as they spread the large capital required to build foundries among their clients. These contract foundries, originally driven by inexpensive labor, have established themselves throughout Asia. Since 1976, the market has dramatically changed, as Asia has increased its share of shipped semiconductors by 60%, according to the Semiconductor Industry Association.

This market shift has caused the US IC industry to change its business model. Large companies such as

because the in-house fabrication process, once monitored very closely, is now being outsourced to potentially untrusted facilities. A recent survey on the semiconductor industry by Semiconductor Equipment and Materials International (SEMI) reports that 90% of companies have experienced IP infringement, with 54% reporting it as serious or extremely serious.¹ Adaptations to the horizontal supply chain are necessary to support the business model and protect the financial and intellectual rights of all the involved parties.

The concerns that arise from IC fabrication fall into three basic categories: metering, theft, and trust. Each category broadly addresses whether extra ICs beyond the purchase order have been produced, whether unauthorized access to information has occurred, and whether the IC has been tampered with.

Although solutions in each category are needed and warranted, our work focuses on metering. We provide an improvement to combinational-locking approaches by recognizing several shortcomings of previous schemes. By increasing synthesis tool flexibility and establishing selection heuristics, we obtain a better security-to-overhead ratio and resiliency in the face of an intelligent adversary.

In this article, we address the need to add reconfigurable-logic barriers to the information flow. We discuss how these barriers can be efficiently implemented, and we suggest the best locations for

them, considering their effectiveness and overhead. We also provide a barrier implementation analysis, including comparisons with other approaches, and we evaluate the resiliency of our approach to attacks.

Our contributions include

- a combinational-locking scheme integrated into a standard CAD tool flow to prevent IC piracy,
- the first metering scheme that does not disclose the entire schematic to the foundry, and
- efficient node selection heuristics for maximizing security while minimizing associated overhead.

The “Related Work on Metering” sidebar discusses other metering approaches for IC supply chain security.

Typical IC fabrication process

Figure 1 shows the process flow for taking a “napkin design” to an IC. The IC designer begins by creating the RTL description of the IC, typically in a hardware description language (HDL). RTL synthesis then involves a series of EDA steps to produce a finalized netlist. First, the logic synthesis stage transforms the RTL design into a directed acyclic graph (DAG), $N = (V, E)$, where N is a logic network, V is a set of logic nodes, and E is the set of edges. The nodes in V represent a Boolean logic function, and the interconnections represent the information flow. Complex heuristics search for graph transformations that optimize the system’s design goals to increase circuit quality. The optimized design is sent to the mapping stage, where the DAG’s Boolean functionality is converted into primitive gates. The placing-and-routing stage searches for the best locations and best connections between each component. Then, the design is exported to a layout-level geometry, in which large plaintext descriptions capture the physical polygons of the IC. This exact blueprint, the layout geometry, is sent to the foundry, which legitimately makes modifications to the polygons to support the design. The fabrication process then continues through its many steps to create, test, package, and distribute the IC.

Our metering approach

Our approach adds reconfigurable-logic barriers to IC prefabrication. These barriers separate the inputs from the outputs such that every path from inputs to outputs passes through a barrier. Figure 2 depicts this approach. The light-gray graphs represent the logic network, with information flowing upward. When the correct key is applied, the information

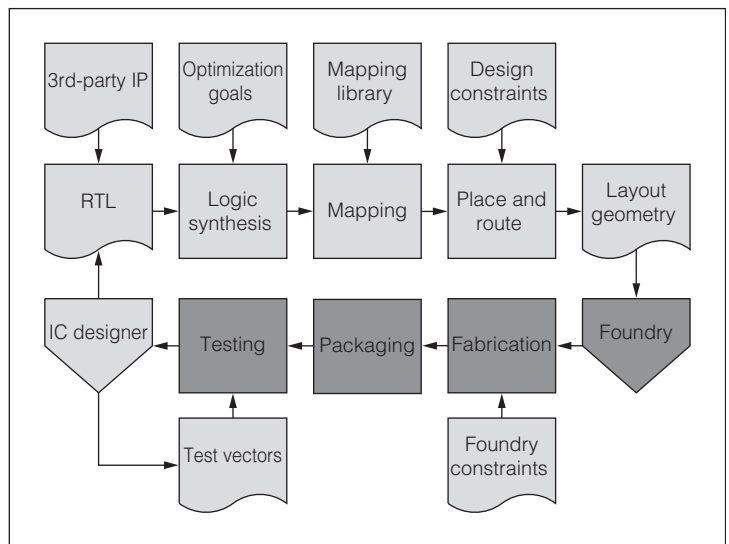


Figure 1. Process flow for IC fabrication, from RTL to IC.

flow is uninterrupted, as indicated in the figure by the straight black arrows. With an incorrect key, the barriers prevent the flow of information, resulting in skewed data flowing out of the barriers, as indicated by the curved black arrows in the figure.

Threat model

Our approach assumes there is a threat of a fabrication attacker external to both the IP creator and the IC designer, which follows directly from the widely popular horizontal contract model. The attacker enters after creation of the layout-level geometry but before IC fabrication. Also, the attacker has access to the layout-level geometry and a set of test vectors, both given by the IC designer. The attacker also has significant resources temporally, fiscally, and computationally, but they are finite. These resources include advanced technical knowledge of IC design and fabrication, access to a foundry, ability to fabricate ICs, and advanced reverse-engineering tools. The attacker’s goal is IC piracy, which is motivated by either profit or acquisition of specialized functionality. In both cases, the attacker’s potential benefit from piracy must outweigh its cost.

IC design partitioning

The standard horizontal IC supply chain is vulnerable in the fabrication phase because the entire IC design and specifications are transferred to the foundry without control over what the foundry will do with them. The foundry is assumed to be well-intentioned so that it will not make extra copies of the IC, steal information, or add a Trojan, but there are no guarantees.

Related Work on Metering

Extensive research has examined IC supply chain security and can be classified into three broad categories: metering, theft, and trust. Because our work addresses metering, we focus here only on previous metering schemes. Metering deals with controlling the number of ICs produced and for whom they are produced. Because of the prohibitively high costs to create foundries, contract fabrication facilities supply the requested number of ICs for a circuit design. Although the foundry might produce the requested number of ICs, the horizontal supply chain does not preclude the foundry from producing additional ICs, which could be sold on the black market. Research on metering tries to let an IC designer control the number of ICs produced or at least the usability of those ICs.

Metering approaches can be either passive or active. *Passive* approaches uniquely identify each IC and register that identity. Later, suspect ICs are checked for proper registration. The uniqueness is usually derived from an unclonable manufacturing variability that is unique for every IC, even those on the same wafer. Both temporal and spatial uniformity are tainted with the inherent parasitics of the IC fabrication process, which decreases yield but is exploited for IC identification. Variabilities such as the threshold mismatch in MOSFET arrays,¹ delay characteristics,² and physical unclonable functions (PUFs)³⁻⁵ are all used to uniquely identify an IC.

Another early work in the passive-metering space explores using control-flow reconfigurability (through a

programmable part) to support many equivalent variable mappings to registers.⁶ Each such mapping serves as the individual signature for a chip's authentication.

Active metering approaches lock each IC until it is unlocked by the IP holder. In the approach proposed by Alkabani and Koushanfar, each IC generates a unique ID on the basis of some spatial variability.⁷ The ICs are initialized to a locked state on power-up, but can be unlocked by the IP holder, who uses the unique ID to generate an unlocking key. Chakraborty and Bhunia added a finite-state machine (FSM) to the netlist, with inputs connected directly to the primary inputs.⁸ The FSM is initially locked and can be unlocked only with the correct sequence of primary inputs. The FSM output is connected to XOR gates dispersed throughout the IC. XOR gate locations are chosen iteratively by greedily selecting nodes with the highest fan-in and fan-out cones.

Our approach also looks at the network's intrinsic properties but uses a combination of better-defined metrics, observability don't-care (ODC) sets, and node positioning. Alkabani, Koushanfar, and Potkonjak enhanced this process by replicating the FSM's states, and they added a key distribution process.⁹ A secondary security measure implemented by Alkabani and Koushanfar augments the FSM with black-hole states, which, when entered, do not allow any outgoing transitions.⁷

A notable active metering approach, EPIC (Ending Piracy of Integrated Circuits),¹⁰ along with a further

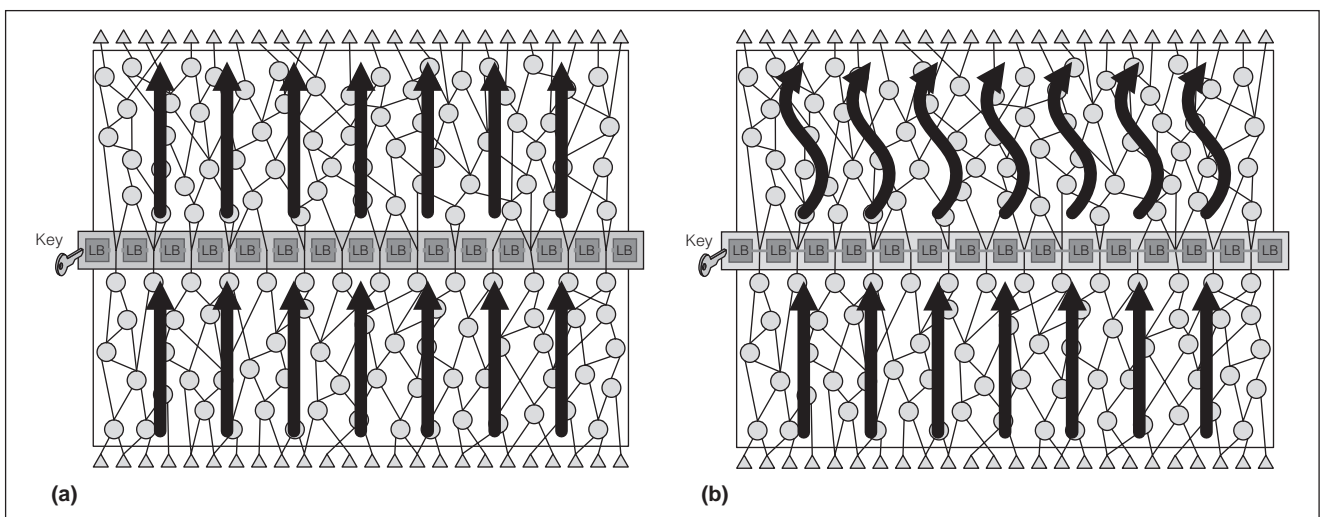


Figure 2. Our metering approach: correct key (a) and incorrect key (b). Logic barriers (LBs) block the information flow when the key is incorrect, but allow it for the correct key.

extension,¹¹ locks the circuit by scattering XOR gates randomly throughout the design. One input to the XOR gate is the original wire; the other is a key bit. With the correct key, the XOR gates act as a buffer; with an incorrect key, they become inverters. A key distribution framework is established using public-key cryptography so that the IP holder, foundry, and individual ICs can all securely participate in the key distribution.

Our approach is similar in terms of the framework into which it fits, but we focus considerable effort on the selection of locking locations rather than randomized ones. By designing more-sophisticated selection heuristics and raising the granularity from XOR gates to look-up tables (LUTs), we not only generalize the key space but also increase the burden on the attacker without increasing overhead. Moreover, our approach does not reveal the entire design to the foundry, thus taking away the foundry's opportunity to reverse-engineer the IC.

References

1. K. Lofstrom, R. Daasch, and D. Taylor, "IC Identification Circuit Using Device Mismatch," *Proc. IEEE Int'l Solid-State Circuits Conf. (ISSCC 00)*, IEEE Press, 2000, pp. 372-373.
2. J. Lee et al., "A Technique to Build a Secret Key in Integrated Circuits for Identification and Authentication Applications," *Proc. Symp. VLSI Circuits*, IEEE Press, 2004, pp. 176-179.
3. S. Kumar et al., "The Butterfly PUF Protecting IP on Every FPGA," *Proc. Int'l Workshop Hardware-Oriented Security and Trust (HOST 08)*, IEEE CS Press, 2008, pp. 67-70.
4. Y. Su, J. Holleman, and B. Otis, "A 1.6pJ/bit 96% Stable Chip ID Generating Circuit Using Process Variations," *Proc. IEEE Int'l Solid-State Circuits Conf. (ISSCC 07)*, IEEE Press, 2007, pp. 406-407, 611.
5. G.E. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," *Proc. 44th Design Automation Conf. (DAC 07)*, ACM Press, pp. 9-14.
6. F. Koushanfar, G. Qu, and M. Potkonjak, "Intellectual Property Metering," *Proc. 4th Int'l Workshop Information Hiding*, LNCS 2137, Springer-Verlag, 2001, pp. 81-95.
7. Y. Alkabani and F. Koushanfar, "Active Hardware Metering for Intellectual Property Protection and Security," *Proc. 16th USENIX Security Symp.*, Usenix Assoc., 2007, pp. 291-306.
8. R.S. Chakraborty and S. Bhunia, "Hardware Protection and Authentication through Netlist Level Obfuscation," *Proc. ACM/IEEE Int'l Conf. Computer-Aided Design (ICCAD 08)*, IEEE Press, 2008, pp. 674-677.
9. Y. Alkabani, F. Koushanfar, and M. Potkonjak, "Remote Activation of ICs for Piracy Prevention and Digital Right Management," *Proc. ACM/IEEE Int'l Conf. Computer-Aided Design (ICCAD 07)*, IEEE Press, 2007, pp. 674-677.
10. J. Roy, F. Koushanfar, and I. Markov, "EPIC: Ending Piracy of Integrated Circuits," *Proc. Design, Automation and Test in Europe Conf. (DATE 08)*, ACM Press, 2008, pp. 1069-1074.
11. J. Huang and J. Lach, "IC Activation and User Authentication for Security-Sensitive Systems," *Proc. Int'l Workshop Hardware-Oriented Security and Trust (HOST 08)*, IEEE CS Press, 2008, pp. 76-80.

We focus our efforts on hindering an attacker's ability to pirate ICs. This problem can be solved by either increasing trust in the foundry or decreasing its ability to pirate. We choose the second option, placing the burden on the IC designer rather than the foundry to enforce.

We have created a scheme that decomposes IP functionality $F(x)$ into F_{fixed} and F_{reconfig} . The majority of the design, F_{fixed} , is given to the foundry to fabricate, but F_{reconfig} (the key in Figure 3) remains with the IP creator.

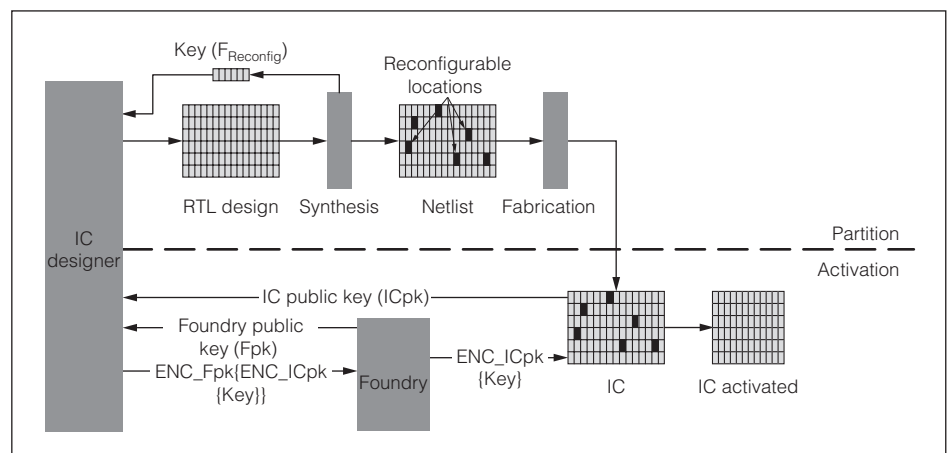


Figure 3. The design is partitioned before fabrication to create a key that is securely distributed to the IC for activation.

Rather than fabricating F_{reconfig} as the original logic, our approach fabricates F_{reconfig} as reconfigurable logic. The withheld F_{reconfig} partition can be programmed into the reconfigurable locations during the activation stage using a secure key distribution framework. An adversarial foundry can make extra ICs; however, without the withheld configuration, the ICs would not function correctly. The foundry could also guess about the configuration, but to make an educated decision regarding the reconfigurable locations, an adversary would have to invest more time and resources than are practical to discover the correct configuration.

Key creation

Two previous combinational-locking schemes use two-input XOR and XNOR gates inserted on random wires throughout the design.^{2,3} We increase the flexibility of combinational locking by using k -input lookup tables (LUTs), and we create selection heuristics to optimize lock placement.

We use LUTs instead of XOR gates for a few reasons. First, in an XOR gate locking scheme, the entire design is given to the foundry. The XOR gates cripple the design for incorrectly guessed key bits but are merely gates added to the design. This means that by reverse-engineering the layout-level geometry, the foundry can obtain the entire IC design. The LUTs not only lock the IC but also replace geometry corresponding to F_{reconfig} with generic memory layout geometry so that only part of the design is transmitted to the foundry. Second, the LUTs have a far larger state space. Each XOR gate requires 1 bit to lock a 1-bit wire, turning it into either a buffer or an inverter. The LUT is exponential in the number of locking combinations for its key bits, thus increasing the attacker's burden.

Because of the key's importance, a distribution framework must be established so that the IP designer can securely unlock each IC. Our research does not create any new methods for distribution, because several popular methods exist. We have chosen to use public-key cryptography, but our work is not limited to this method. Each IC uses physical unclonable manufacturing variability to differentiate itself from other ICs on the same wafer and to create a unique public-key encryption pair.⁴⁻⁸ The transmission of F_{reconfig} from the IC designer to the IC is secured, and ICs can be activated individually.

Selection heuristic

Controllability and observability are two inherent metrics of a logic network. We use these metrics for part of our selection heuristic, expressed as don't-care conditions. Informally, these metrics constitute a set of inputs or outputs that do not matter to the circuit, because of the circuit and logic topology. *Controllability* measures how easy it is to control a node's inputs or how easy it is to generate a specific set of inputs for that node. *Observability* measures how easy it is to see a particular set of values at internal, inaccessible nodes on the primary, observable outputs. These metrics are often expressed in terms of don't-care minterm sets CDC_{in} (controllability don't-care) and ODC_{out} (observability don't-care).

Another parameter in our heuristic is the height of a node. For the logic network $N = \{V, E\}$, each node, $v_i \in V$, has an associated height equal to the length of the shortest simple path starting at the primary inputs. Therefore, all primary inputs have a height of 0, and one of the primary outputs has the maximum height.

Moreover, our selection heuristic uses a cut-based approach. A *cut* is a network partition through which every path from a primary input to a primary output passes. By factoring cuts into our heuristic, we can choose nodes globally rather than through a greedy selection. Complete enumeration of the cuts is not possible, because the logic network grows exponentially. Consequently, we traverse the network in topological order, advancing each cut one node at a time.

When choosing locations to insert the reconfigurable regions representing F_{reconfig} , our goal is to minimize the possibility of the attacker's bypassing the locks or guessing the correct configuration. So, we select the best cut of the network, considering the ODC set and cut height (see Figure 4).

We use ODC because, intuitively, if F_{reconfig} is important, any contamination in F_{reconfig} is observed as a contamination in F with as high a probability as possible. When selecting F_{reconfig} , it is best to choose nodes that maximize control by being highly observable over outputs $(y_0, y_1, \dots, y_{N-1})$. This means a change in the node will affect the output more than a less-observable node. Formally, for the logic network $N = \{V, E\}$, corresponding to a function $f(x_0, x_1, \dots, x_{n-1}) = (y_0, y_1, \dots, y_{N-1})$ with the desired threshold of reconfigurable logic, $0 \leq TH_{\text{reconfig}} < 1$, we choose subset $V_{\text{reconfig}} \subseteq V$ such that $|V_{\text{reconfig}}|/|V| \leq TH_{\text{reconfig}}$ and the observability of subunits G_i


```

for all  $n_i \in nodes$  do
    Calculate ODC set of  $n_i$ 
    Calculate height of  $n_i$ 
end for
for all  $n_i \in |nodes|$  in topological order do
    Add  $n_i$  to cut
    Trim children of  $n_i$ 

    Cutscore[ $n_i$ ] =  $\alpha \times \left| \frac{\max Height - 2 \times cutHeight}{\max Height} \right|$ 
                    +  $(1 - \alpha) \times \left( \frac{cutODCSum}{\max ODCSum} \right)$ 

end for
Select best cut C
for all  $n_i \in C$  do
    Tag  $n_i$ 
end for

```

Figure 4. Algorithm that selects the best network cut (partitioning) for our selection heuristic.

in f , $\sum_{G_i \in V_{reconfig}} \sum_{j=0}^{N-1} \min(x_0, x_1, \dots, x_{N-1}) \left(\frac{\partial v_i}{\partial G_i} \right)$, is maximized.

We also use height as a metric for our selection criteria. Specifically, cuts near the center of the logic network are better than cuts deviating in either direction. Cuts near the primary outputs are highly observable but not very controllable, and cuts at the primary inputs have the opposite characteristics. Highly controllable nodes are easy for an attacker to exercise, and not very observable nodes restrict the lock's effectiveness. The nodes in the middle of the network blend both controllability and observability, helping to increase the attacker's burden.

To properly mix the ODC minterm score with the height value, we introduced the parameter α to tune the cut selection's dependence on the metrics. An α value of 0 considers only the ODC score; an α value of 100 relies completely on the height.

Implementation

Figure 5 shows the overall tool flow for implementing our heuristic. Light-gray boxes represent unmodified steps in the traditional tool flow; light-gray boxes with diagonal lines represent the additional integrated steps. Dark-gray boxes represent the specific tools used for those phases. On the left side of the figure, the IC designer creates the RTL design. Logic synthesis begins with logic optimization, which optimizes for literal count. Next, the ODC and height values are calculated for each node. The cuts are enumerated and ranked on the basis of the selection heuristic scoring, the best cut is selected, and the cut's nodes are tagged. Tagging lets the mapping phase discern between traditional primitive gates and reconfigurable logic. Once the mapping phase completes, a netlist is generated for simulation. From this point, the process can continue through placing and routing, before generating a layout geometry file to be used by the foundry for fabrication.

Experimental analysis

Here, we provide the results obtained from implementing our approach in the following experiments. We compare the cut-based selection heuristic with random selection, explore the effectiveness of a

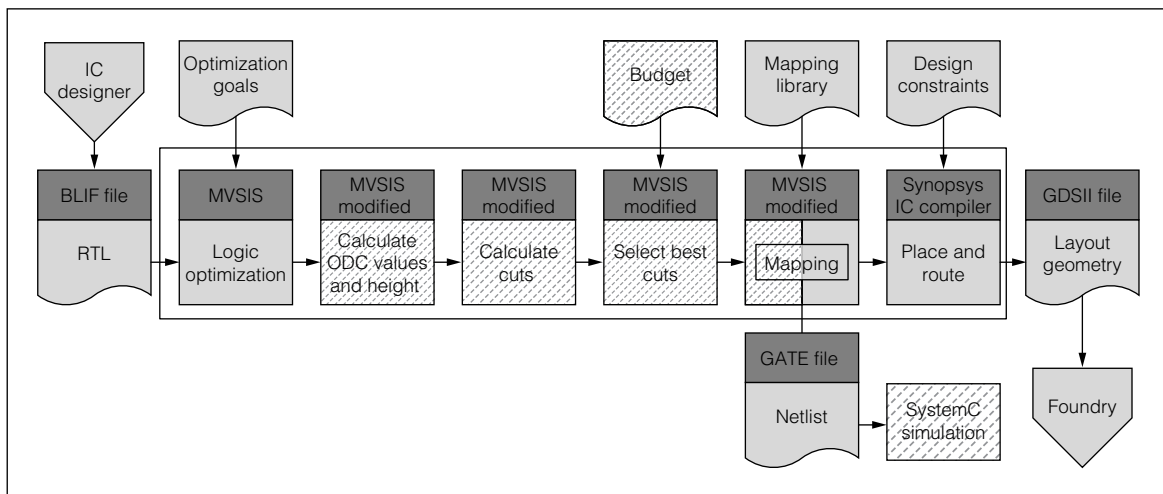


Figure 5. Tool flow used to create the layout geometry from the RTL description.

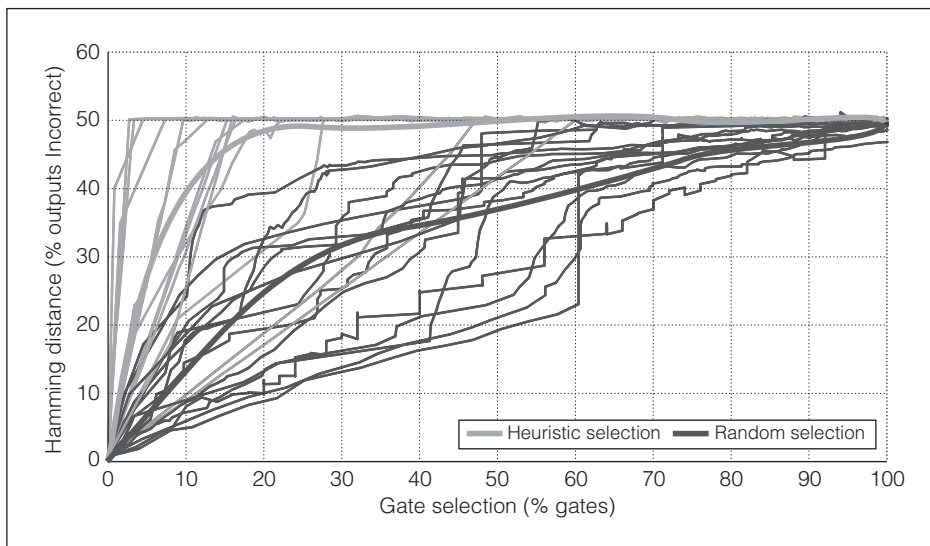


Figure 6. Heuristic selection vs. random selection for all benchmarks.

single cut, and evaluate the performance effects from tuning our heuristic's parameters. The favorable results indicate that our tuned selection heuristic is more resilient to attacks than previous methods while minimizing the overhead induced on a variety of professional benchmarks. The results also highlight the feasibility of integrating security features into the IC design flow aimed at decreasing IC piracy.

All of the benchmarks used in the experiments were part of the combinational multilevel subset of the Microelectronics Center of North Carolina (MCNC) benchmark suite, which is popular in logic synthesis literature. We implemented our approach in a standard tool flow, as shown in Figure 5. The logic synthesis was provided by MVSIS (Multiple Valued Sequential logic Interactive System), an open-source tool from the University of California, Berkeley. The results fed into three custom steps added to the source code of MVSIS. These included calculating each node's ODC and height value, calculating the cuts, and tagging the best nodes. We used the Synopsys IC Compiler for placing and routing the circuit to generate a layout geometry. In parallel, we performed SystemC simulations on the circuit (after mapping), to observe the circuit's functionality.

Heuristic selection vs. random selection

In previous works,^{2,3} researchers chose nodes randomly to be replaced by XOR gates, whereas our work used selection heuristics. In this experiment, we compared the effects of random and heuristics selection. For each benchmark, we generated 202 netlists by

selecting percentages of nodes from 0% to 100% in increments of 1%, and then generating one netlist with random selection and one with the heuristic-based selection for each percentage. We assumed there was a brute-force attacker who randomly configured the LUTs and knew the expected outputs for the applied test vectors. We simulated each benchmark with millions of test vectors and calculated the average Hamming distance over all outputs and test vectors.

Because we want to obscure the logic network so that an attacker cannot gain the LUT configurations, a large Hamming distance, approaching 50% to mimic random logic, is desirable, with a smaller percentage of nodes selected to minimize the overhead. Figure 6 shows the results of this experiment. Dark-gray lines represent random selection; light-gray lines represent heuristics selection. Thin lines represent individual benchmarks; thick lines represent trend lines that fit those benchmarks. The heuristic selection lines rise to 50% far more quickly than the random selection lines, showing that it takes less overhead for an intelligent selection of nodes to increase the attacker's burden.

Single-cut selection

In the second experiment, we created a single netlist for each benchmark, representing the percentage of gates selected to complete a single cut through the network. Figure 7 shows how the benchmarks are affected by a single-cut heuristic. For each benchmark, the gray bar chart represents the Hamming distance on the left vertical axis, and the line graph represents the percentage of gates in a single-cut selection on the right vertical axis. The Hamming distances for a single cut were very good, and for every benchmark were within 0.1% of being at 50%. The percentage of gates selected was also very low for most of the benchmarks (an average of 8.08%). This means that a single cut is sufficient to make the circuit act as random logic while only requiring a small percentage of reconfigurable logic. For all of the following experiments, the heuristic-based selection used a single cut.

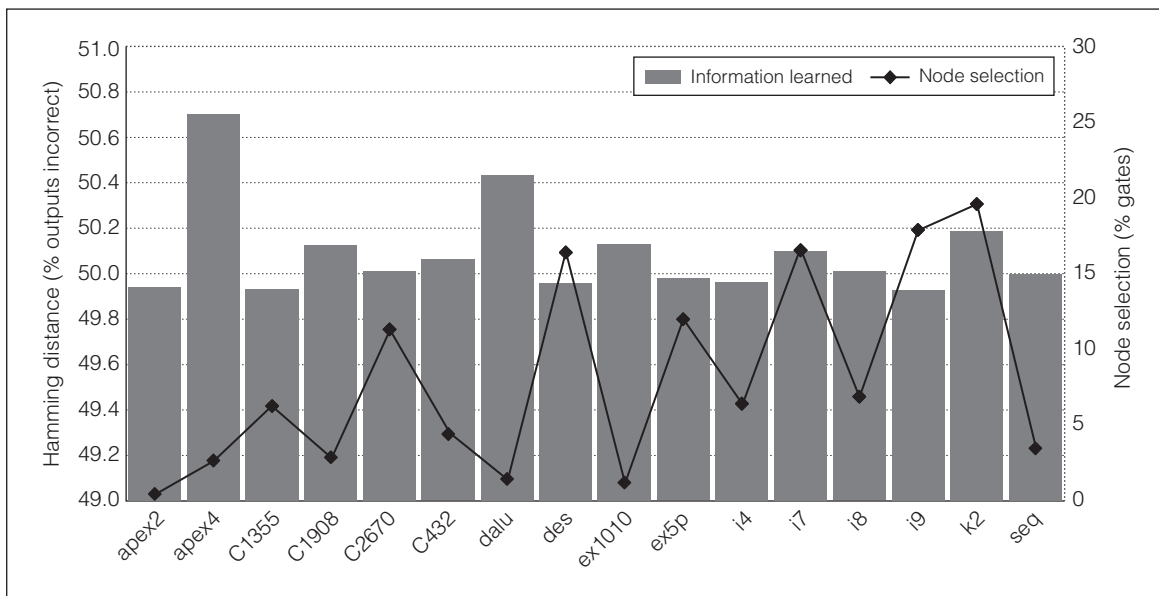


Figure 7. Hamming distance and overhead for a single cut.

Cut height

This experiment empirically tested the effects of using a cut's height in the selection heuristics. We added parameter α (explained earlier) to weight the heuristic's dependence on ODC and height values. For the previous experiments, we set α to 0 so that the height had no effect on the cut score. But, for this experiment, α varied from 0 to 100.

At each α value, we generated a netlist using the single best cut. Because α is a weighting parameter, less than 101 unique netlists were created, since two different α values can choose the same cut. We simulated each unique netlist, assuming an intelligent attacker rather than a brute-force attacker. The attacker measured the percentage of correct outputs for each test vector and treated the LUTs as independent rather than as one large key space. We held the LUT configurations constant for all LUTs except the one that was iterated over every configuration. For each configuration, we applied millions of test vectors and monitored the outputs. The configuration leading to the greatest amount of information gained was chosen. Then, with this value fixed, the next LUT was targeted. This process continued through all the LUTs and then back around to the first one until 400,000,000 LUT configurations were tried.

Because we want to minimize the possibility of an attacker's figuring out the key, in our experiment we chose nodes for LUT replacement that minimized the amount of information an attacker

could learn. Figure 8 shows how α affects the amount of information learned for a single benchmark, Apex4, as the effort increases, with each series representing a unique α value. Every series plateaus at a percentage of information learned; then, no more information is gained as the effort continues to increase. The α values closer to 50 plateau at a lower percentage compared to the α values closer to the extremes. Therefore, an approximately equal weighting between the ODC and height results in the largest burden for an attacker because he or she does not learn as much information. Also, because no more information is learned as the effort increases, this approach is resilient to an attacker.

Figure 9 aggregates the data from all the benchmarks. Thin lines represent a single benchmark; the thick line represents the average of all benchmarks. With an α value of 0, the attacker can gain about 90% of the information. But, as α approaches 50, far less information can be obtained. Finally, as α continues toward 100, more information is obtained. This result implies that an α value near 50 decreases an attacker's ability to learn about the circuit.

In these experiments, our node selection heuristic outperformed previous approaches' random selection. Separating the logic network with a single cut yields the best results for the associated overhead. Also, by introducing an extra parameter, α , the quality of the selection was further increased.

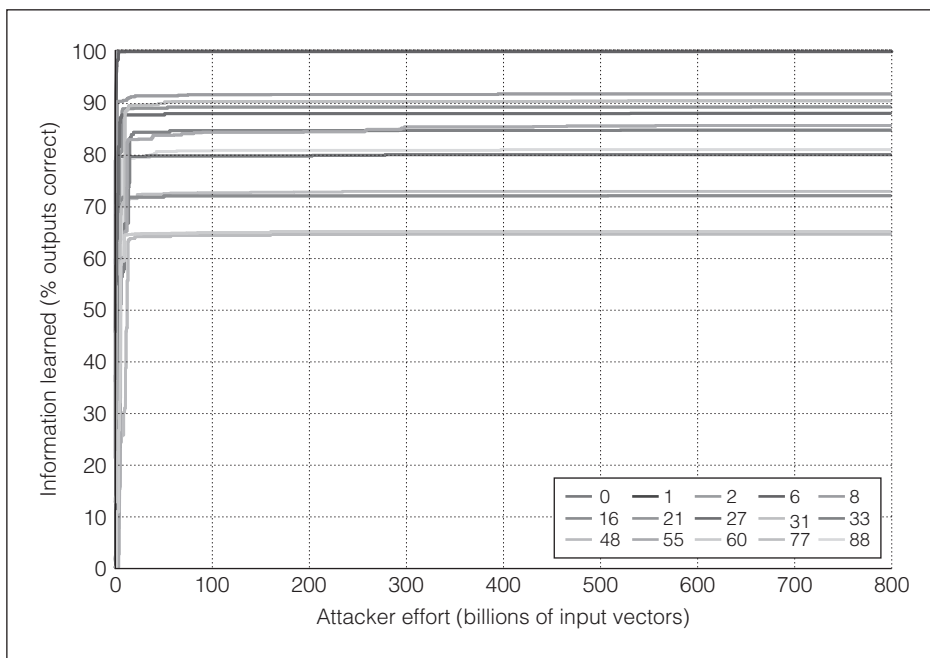


Figure 8. Percentage of information learned as a function of the number of input vector guesses. Series names correspond to the α value that led to the cut selection.

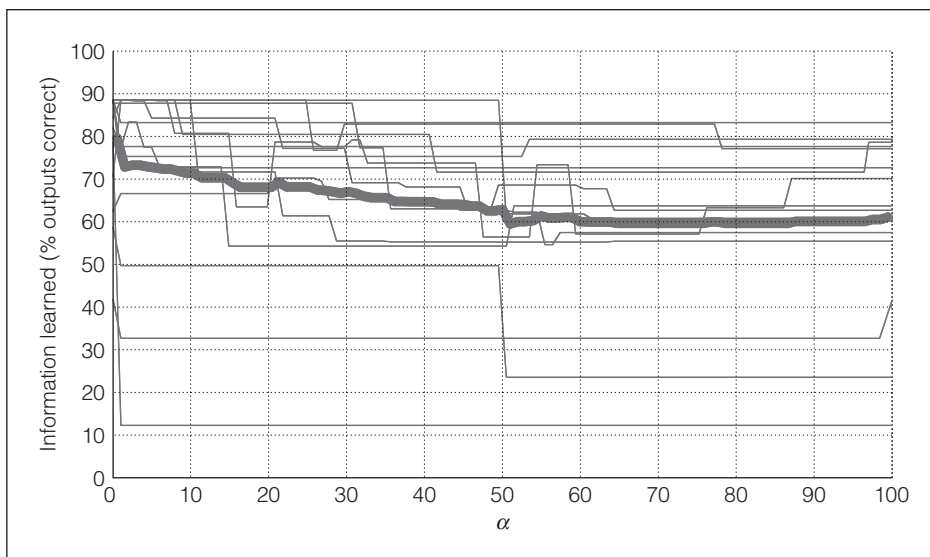


Figure 9. Percentage of information learned as a function of α for all benchmarks.

OUR FUTURE WORK in this area is expected to proceed along three main avenues. First, we intend to more accurately model the tools and techniques available to an adversary at a foundry. Second, we are looking into providing a more in-depth analysis of the VLSI cost of our approach, in terms of chip area, performance, and power consumption. Taken together, these efforts will

allow us to more accurately measure the security-overhead trade-off inherent in our specific approach, and in combinational locking schemes in general. Finally, we intend to explore how our reconfigurable logic barriers can be integrated into traditional semiconductor testing and verification approaches, so that valid chips can be identified at the fab without revealing secret-key information. ■

References

1. "Intellectual Property (IP) Challenges and Concerns of the Semiconductor Equipment and Materials Industry," white paper, Semiconductor Equipment and Materials Int'l, Apr. 2008; <http://www.semi.org/eu/Issues/PublicPolicy/ssLINK/P043701>.
2. J. Huang and J. Lach, "IC Activation and User Authentication for Security-Sensitive Systems," *Proc. Int'l Workshop Hardware-Oriented Security and Trust (HOST 08)*, IEEE CS Press, 2008, pp. 76-80.
3. J. Roy, F. Koushanfar, and I. Markov, "EPIC: Ending Piracy of Integrated Circuits," *Proc. Design, Automation and Test in Europe Conf. (DATE 08)*, ACM Press, 2008, pp. 1069-1074.
4. S. Kumar et al., "The Butterfly PUF Protecting IP on Every FPGA," *Proc. Int'l Workshop Hardware-Oriented Security and Trust (HOST 08)*, IEEE CS Press, 2008, pp. 67-70.
5. J. Lee et al., "A Technique to Build a Secret Key in Integrated Circuits for Identification and Authentication Applications," *Proc. Symp. VLSI Circuits*, IEEE Press, 2004, pp. 176-179.
6. K. Lofstrom, R. Daasch, and D. Taylor, "IC Identification Circuit Using Device Mismatch," *Proc. IEEE Int'l Solid-State Circuits Conf. (ISSCC 00)*, IEEE Press, 2000, pp. 372-373.

7. Y. Su, J. Holleman, and B. Otis, "A 1.6pJ/bit 96% Stable Chip ID Generating Circuit Using Process Variations," *Proc. IEEE Int'l Solid-State Circuits Conf. (ISSCC 07)*, IEEE Press, 2007, pp. 406-407, 611.
8. G.E. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," *Proc. 44th Design Automation Conf. (DAC 07)*, ACM Press, pp. 9-14.

Alex Baumgarten is a software design engineer in test at Microsoft in Redmond, Washington. He performed the work described in this article while pursuing graduate studies in the Department of Electrical and Computer Engineering at Iowa State University. His research interests include reconfigurable computing, embedded systems, and hardware security. He has an MS in computer engineering from Iowa State University.


Akhilesh Tyagi is an associate professor of electrical and computer engineering at Iowa State University. His research interests include computer architecture,

compilers, trusted computing platforms, and VLSI complexity theory and low-energy design. He has a PhD in computer science from the University of Washington, Seattle.

Joseph Zambreno is an assistant professor of electrical and computer engineering at Iowa State University. His research interests include computer architecture and compilers, reconfigurable computing as a general enabling technology, and the use of design automation to address various aspects of security and trust. He has a PhD in electrical and computer engineering from Northwestern University.

■ Direct questions and comments about this article to Joseph Zambreno, Department of Electrical and Computer Engineering, Iowa State University, 2215 Coover Hall, Ames, IA 50011; zambreno@iastate.edu.

cn Selected CS articles and columns are also available for free at <http://ComputingNow.computer.org>.



COMPUTING LIVES

www.computer.org/annals/computing-lives

The "Computing Lives" Podcast series of selected articles from the *IEEE Annals of the History of Computing* cover the breadth of computer history. This series features scholarly accounts by leading computer scientists and historians, as well as firsthand stories by computer pioneers.