A Physical Design Flow Against Front-Side Probing Attacks by Internal Shielding

Huanyu Wang, *Student Member, IEEE*, Qihang Shi, *Member, IEEE*, Adib Nahiyan, *Student Member, IEEE*, Domenic Forte, *Senior Member, IEEE*, and Mark M. Tehranipoor, *Fellow, IEEE*

***Abstract*—Security-critical applications on integrated circuits (ICs) are threatened by probing attacks that extract sensitive information assisted with focused ion beam (FIB)-based circuit edit. Existing countermeasures, such as active shield, analog shield, and *t*-private circuit, have proven to be inefficient and provide limited resistance against probing attacks without taking FIB capabilities into consideration. In this article, we propose an FIB-aware anti-probing physical design flow, which considers FIB capabilities and utilizes computer-aided design (CAD) tools, to automatically reduce the probing attack vulnerability of an IC's security-critical nets with minimal extra design effort. The floor-planning and routing of the design are constrained by incorporating three new steps in the conventional physical design flow, so that security-critical nets are protected by *internal shield* nets with low overhead. Results show that the proposed technique can reduce the vulnerable area exposed to probing on security-critical nets by 100% with all critical nets fully protected for both advanced encryption standard (AES) and data encryption standard (DES) modules. The timing, area, and power overheads are less than 3% per module, which would be negligible in a system-on-chip (SoC) design.**

***Index Terms*—Computer-aided design (CAD), focused ion beam (FIB), hardware security, probing attack, very large-scale integration (VLSI) physical design.**

I. Introduction

IN LIGHT of the increasing performance of integrated circuits (ICs), society's reliance on these electronic computing systems is deepening. Meanwhile, various software and hardware-based attacks are threatening the integrity and confidentiality of security-critical information stored in ICs, for instance cryptographic keys, firmware, communication credentials, device configuration, and private data. Solutions to protect against cyber and noninvasive physical attacks, such as buffer overflow and side channel analysis, have been widely explored; however, there is no sufficient investigation on countermeasures against physical probing attacks. In a probing attack [1], the internal wires of security-critical IC devices, such as smart cards, smart phones, military systems, and financial systems, are physically tapped to extract sensitive information. Even if the design is equipped with protection mechanisms, an attacker is still likely to bypass the protection and expose the signal nets carrying security-critical information through focused ion beam (FIB) systems [2]. FIB is a powerful circuit editing tool that can mill and deposit material on silicon dies with sub-10-nm level precision [3], [4]. Note that FIB's resolution is keeping pace with technology scaling; further, an attacker does not need to purchase a new one. Instead, FIBs are available to rent or purchase second-hand at low cost. In the Internet of Things (IoT) era, the threat from probing is aggravated since there will be a larger volume of low-end devices which are physically accessible.

In recent literature, various countermeasures, e.g., active shield [5], [6], analog sensors [7], [22], and *t*-private circuit [8], could be utilized to protect security-critical circuits against probing attacks. *Active shield* is the most common method, which detects milling by placing a dynamic signal carrying wire mesh as a protective shield on the top-most metal layer of the chip [5], [6], against front-side probing attacks which occur from the passivation layer and through upper metal layers. To detect the attack, a digital pattern is transferred through the shield wires, and the received signals are compared with the same pattern from the lower metal layers. If a mismatch at the comparator is detected, an alarm will be triggered, which results in a security action, such as erasure of sensitive information or shut-down of the device. Unfortunately, large area and design overhead and routing congestion are imposed on the design by active shield. Further, it can be easily disabled or bypassed by FIB's circuit edit capability as demonstrated in [11]–[13], [18], and [19]. *Analog sensors*, which measures analog parameters of the victim wires, such as capacitance and delay to detect the attack, can be an alternative approach to active shield. However, the main challenge for analog sensors is the low reliability due to process variation in advanced technology nodes. Weiner *et al.* [22] recently proposed CaLIAD to compensate the manufacturing variations. Ishai *et al.* [8] proposed the *t-private* circuit approach where a security-critical circuit is transformed so that at least $t + 1$ probes are required within one clock cycle to extract 1-bit information. Though *t*-private

Manuscript received April 25, 2019; revised July 15, 2019 and October 1, 2019; accepted October 13, 2019. Date of publication November 7, 2019; date of current version September 18, 2020. This work was supported in part by NSF under Project 1717392, in part by SRC under Grant 2769.001, and in part by AFOSR MURI under Project FA9550-14-1-0351. This article was recommended by Associate Editor C. H. Chang. *(Corresponding author: Huanyu Wang.)*

The authors are with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611 USA (e-mail: huanyuwang@ufl.edu; qihang.shi@ufl.edu; adib1991@ufl.edu; dforte@ece.ufl.edu; tehranipoor@ece.ufl.edu).

Digital Object Identifier 10.1109/TCAD.2019.2952133

circuit increases the probing attack's difficulty and time cost, its $O(t^2)$ times area overhead for design transformation is prohibitively expensive [9].

Existing countermeasures are ad hoc with inefficient protection, are not designed to counter FIB-based attack, and require prohibitive area and design overhead [9]. Further, there is no holistic and efficient approach that can be easily incorporated into conventional application-specific IC (ASIC) design flow to protect security-critical circuits and nets from probing attack. Therefore in this article, we make the following major contributions to mitigate front-side probing attack.

1) A highly automated physical layout design flow that mitigates the threat of front-side probing attacks and is easy to integrate into existing electronic design automation (EDA) design flows.
2) An internal shield design that is not limited to top layer and full-die area shapes, and does not require extra pattern generator circuit, which dramatically reduces area and routing overhead. Because the shield is placed on an internal layer rather than the top layer, the shield is far more difficult to reverse engineer, bypass, or reroute.
3) Instead of dedicated pattern generators, the proposed shield design uses nets from existing functional design, selected by a shield net identification metric. A method is also developed to choose the best two layers for multilayer shield designs based on the technology specifications, which can provide better protection to security-critical nets than single layer shield.
4) A metric is developed to identify security-critical nets that are most likely to be targeted for probing attacks, thus enabling a shield design that does not have to cover entire die area. Such nets include those directly connected to the security asset as well as nets in the asset's fanout from which sensitive information could be derived.
5) The proposed approach is evaluated on advanced encryption standard (AES) and data encryption standard (DES) modules using the exposed area (EA) metric proposed in [13]. Results show that the vulnerable area exposed to probing attacks decreases from 80% without shield to zero for FIB aspect ratio smaller than 5, i.e., all security-critical nets are completely protected or covered in AES and DES. The overhead is less than 3% for timing, power, and area.

The rest of this article is organized as follows. In Section II, we provide background on probing attacks and related research. In Section III, we present our probing-aware design flow, including target/shield identification, shield layer selection, constrained layout, and EA calculation. The evaluation results are provided in Section IV. Finally, we conclude in Section V.

II. Background

A. Assets

An *asset* is an information resource worth protecting from extraction by a would-be adversary [10]. Compromise of assets could cause tremendous damage to intellectual properties (IPs), digital privacy, and digital rights management. Examples of assets that are likely targeted in a probing attack include the following [9], [10].

1) *Keys:* Private keys used for encryption/decryption operations.
2) *Firmware and Bitstream:* Instruction codes of microprocessors and configuration bitstream of field programmable gate arrays.
3) *On-Device Protected Data:* Sensitive information such as financial data, personal health information, passwords, etc. stored on chip.
4) *Device Configuration:* Configuration data that controls the access permission of a device or a module.
5) *Cryptographic Random Number:* Random numbers generated as keys, one-time pads, and initialization vectors.

B. Typical Probing Attack Steps

It is vital that assessment of a protective design should be performed with full knowledge about the attack it intends to prevent. Typical probing attacks [9] consist of these following fundamental steps, all of which must be successful for the attack to succeed.

1) Decapsulation.
2) Reverse engineering of the chip under attack.
3) Locating the target wires.
4) Exposing the target wires to probes.
5) Extracting target information from signals collected with the probe.

To expose the chip die, the chip package needs to be partially or fully removed depending on the occupied area of attacker's probing target wires. This is the first stage of most invasive physical attacks, which requires sufficient practice handling noxious chemicals, such as fuming nitric acid at 60 °C combined with acetone to remove plastic packages [1]. The attacker can also remove the copper plate mechanically from the back-side to decapsulate the chip without chemical etching.

Next, detailed design information could be extracted through reverse engineering [2], which is the iterative process of delayering and imaging to figure out the structure and functionality of the chip. Identifying the assets nets, such as encryption keys, is one of the most important job for the step of reverse engineering in the case of probing attack. The probing target wires' (asset nets) locations can then be figured out by one-to-one correspondence between the netlist and layout. Reverse engineering can also help determine whether the cut of a wire would impact the asset extraction or not. Advanced automatic tools, such as ICWorks from Chipworks [16], pix2net from [20], and ChipJuice from Texplained [21], can perform netlist extraction automatically from images of each layer captured through optical or scanning electron microscopes (SEMs), which accelerates the reverse engineering process to a great degree.

After the probing target nets have been identified by reverse engineering step, the next step is physically locating the metal wires associated with the target nets on the IC under

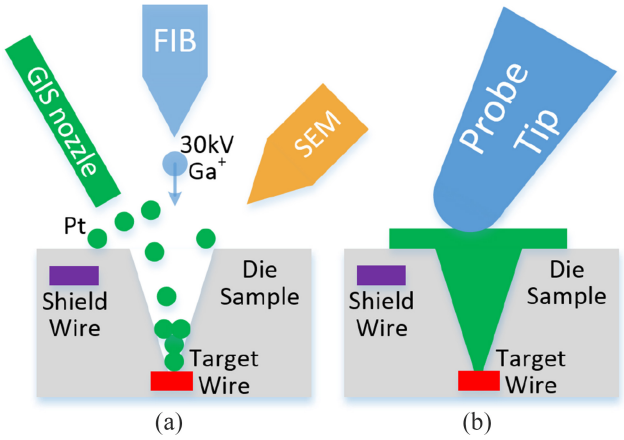


Fig. 1. (a) FIB deposits platinum in the milling cavity to build conducting path from target wire. (b) Deposited conducting path serves as electrical probe contact.

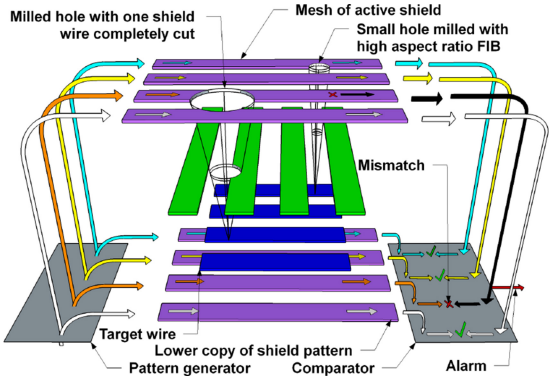


Fig. 2. Basic working principle of active shield and bypass attack on active shield.

attack. The main challenge of this step is that although the attacker has located the probing target wires on the sacrificial chips during reverse engineering process, the target coordinates obtained from previous experience may not guarantee the success of the attack because of the FIB kinetic errors. Further, the attacker has to find the coordinates of the point to mill at blindly on the chip under attack, because the attacker cannot expose anything on the targeted device beyond absolute necessity to help him locate target wires, which requires a precise-enough kinematic mount, and fiducial markers (i.e., visual points of reference on the device) to base these coordinates.

When the probing target wires are located on the chip under attack, the next step is to expose the target wire and build a conducting path for probing without damaging any other parts of the circuitry, e.g., wires or vias surrounding the target wire, on the chip. Modern FIB systems, such as ZEISS ORION NanoFab which can edit out obstructing circuitry with 5-nm level precision, may be used to accomplish this step. First, a cavity is milled on the chip to expose target wires on lower layer as shown in Fig. 1(a). Then, the gas injection system (GIS) nozzle installed at the chip front-side surface will release platinum (Pt) or tungsten (W) gas, whose atoms could be deposited in the milling cavity to build a conducting path that can serve as electrical probe contacts under the help of high energy ion beam as shown in Fig. 1(b). FIB aspect ratio is an important parameter for FIB system which is defined as the ratio between the depth and diameter of the milling hole. The larger of the aspect ratio, the narrower of the hole. Modern FIB system, like ZEISS ORION NanoFab, can reach the aspect ratio of $\sim$30 for milling. However, for depositing, the metal gas atoms cannot get to the bottom of the milling hole if the hole is too narrow because these atoms may block the hole in the middle. So, a reasonable maximum aspect ratio for depositing is $\sim$10 which is also the maximum FIB aspect ratio considered in this article. The high resolution of modern FIB systems implies that many probing attack countermeasures can be eliminated by simply cutting a few wires. Note that an FIB equipped attack can typically place no more than 8 simultaneous probes to inject signals by a function generator or capture signals by a logic analyzer.

The last step of probing attack is to extract the asset signal. As long as the asset wires are properly exposed and connected to the conducting path without triggering any probing alarms from active or analog shields, the asset signals can be extracted using a probing station. There are few difficulties for this step. First, some software and hardware processes might need to be synchronized and completed before the asset is available. Further, the asset information may only exist for a very short period, e.g., only few clock cycles. In addition, if the chip has an internal clock source to prevent external manipulation, the attacker will need to either disable it or synchronize his own clock with it.

Each step can have a number of alternative techniques where success with only one of them is necessary. For example, locating target wires in layout can be done by reverse engineering the design or with information from a similar IP core. Obfuscation can force the attacker to spend more time on this step, but if the IP is reused in another design it would allow attacker to circumvent it.

C. Countermeasures and Limitations

In this section, we briefly review existing countermeasures against probing attacks, such as active sensor, analog sensor, charge sensor, and *t*-private circuit, and list their limitations.

Active shield is so far the most widely used countermeasure against front-side probing attack. In this technique, a shield carrying signal patterns is placed on the top-most metal layer of the chip to detect FIB tampering. As shown in Fig. 2, a digital pattern is generated from a pattern generator, transmitted through the shield wires on top-most metal layer (upper purple), and then compared with a copy of itself transmitted from lower layer (lower purple). If an attacker mills through the shield wires, a hole is expected to cut one or more shield wires, thereby leading to a mismatch at the comparator and triggering an alarm to erase or stop generating sensitive information. Despite its popularity, the biggest problem for active shield is that they impose large design overheads, and are very vulnerable to attacks with advanced FIBs. Aspect ratio is a measure of the FIB performance defined as the ratio between milled

hole depth and diameter [4]. A FIB with high aspect ratio can penetrate the shield with a hole of smaller diameter by leveraging the space between shield wires without damaging the shield wires, which is called *bypass attack*. FIB's depositing capability allows the attacker to implement *reroute attack* which makes the shield wire free of cutting by rerouting a copy path between identified equipotential points [11], [12]. When milling and depositing in nanometer scale and applied on silicon ICs, state-of-the-art FIB systems can reach an aspect ratio up to ∼10 [3]. Another problem with active shield method is that at least an entire metal routing layer must be dedicated to the shield, which does not go well with designs with tight cost margin, or designs with few routing layers.

An alternative approach to active shield is to construct an analog shield or sensor. Analog sensors monitor parametric disturbances, such as capacitance, RC delay, with the victim wires. The probe attempt detector (PAD) [7] uses capacitance measurement on selected security critical wires to detect additional capacitance introduced by a metal probe. Compared to active shields, analog sensors require less area overhead. The problem with analog sensors or shields is that analog measurements are less reliable due to process and environmental variations, a problem further exacerbated by feature scaling. Weiner *et al.* [22] recently proposed CaLIAD to compensate the manufacturing variations.

The t-private circuit technique is proposed in [8] based on the assumption that the number of concurrent probe channels that an attacker could use is limited, and exhausting this resource thereby deters an attack. In this technique, the circuit of a security-critical block is transformed so that at least $t + 1$ probes are required within one clock cycle to extract one bit of information. The major issue with t-private circuit is that the area overhead involved for the transformation is prohibitively expensive ($O(t^2)$). In addition, generating and protecting the random signals from being disabled by FIB is nontrivial.

D. Threat Model

In this article, we restrict our focus on electrical probing from the front-side, although our proposed technique also applies to back-side attacks that occur through the silicon substrate rather than top-level passivation targeting low-layer interconnects. Back-side optical probing and attacks targeting gate diffusion silicides are based on an entirely different mechanism and therefore out-of-scope of this article, which will likely need another future work to address. The objective of the adversary this article intends to deter is to extract assets stored in a device through probing attacks with advanced circuit edit tools such as a FIB. We further assume a strong attacker that has full layout information of the design from either reverse engineering or bribing a rogue employee in the foundry. The attack is assumed to be performed by milling a cavity to expose the sensitive net, depositing a conductor in the cavity to build a contact pad on chip surface, and probing at the pad to extract sensitive information.

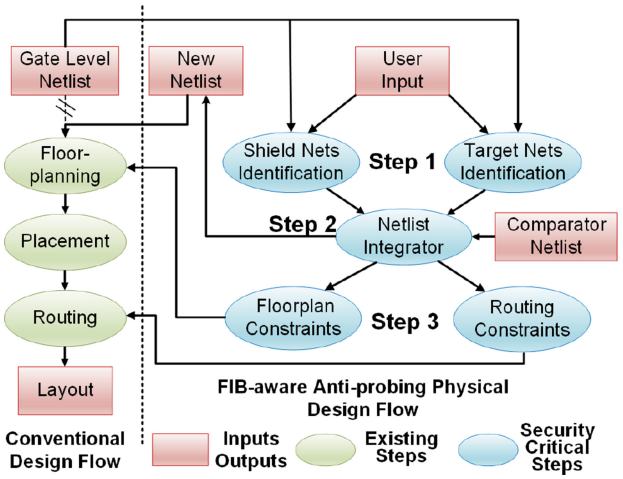


Fig. 3. Overall FIB-aware anti-probing physical design flow.

III. Anti-Probing Design Flow

Our objective is to develop an FIB-aware anti-probing physical design flow that incorporates automated security-aware floor-planning, cell placement, routing, and evaluation into the conventional flow in order to protect the security-critical nets against front-side probing attacks. We shall accomplish this by using a chip's internal functional nets as "shield" nets on upper layers to provide coverage for "target" nets (i.e., those carrying asset signals) on lower layers in the design. Another copy of shield nets will be routed in lower layers. Once at least one shield net on upper layer is cut off in an attack, the comparator will detect the mismatch between the signal on upper shield net and the one from lower layer. An alarm will be triggered to take the appropriate actions (e.g., terminate the operation of the chip or remove all asset information). Note that by leveraging the internal functional nets of the design itself for protection without adding any extra large circuitry, like the pattern generator and shielding circuit in active shield approach, the overhead of our approach is very low. In addition, when shield nets are placed within internal metal layers, they will be far more difficult for an attacker to bypass and reroute than dedicated shields, like active shield which typically resides at top metal layer, since the metal wires above the shield layer will be a huge obstacle to circumvent during the attack. Also, our approach can complement those PAD-based techniques, e.g., PAD can protect buses efficiently while our approach can protect PAD and other nonbus circuits. Further, the whole anti-probing physical design flow is implemented using computer-aided design (CAD) tools, which means the whole process could be completely automatic and uniform for different designs so that the design overhead to build the proposed internal shield will be very limited.

The overall workflow of our anti-probing physical design flow is shown in Fig. 3. It brings into conventional ASIC design flow three new steps. First, appropriate shield nets and target nets are identified for optimal protection against probing attack. Sections III-A and III-B will illustrate the detailed requirements and metrics to identify target nets and shield nets, respectively. User input includes asset information and

threshold values to identify target nets and shield nets. Then, a comparator is inserted in the gate-level netlist of the original design to detect the mismatch. The comparator itself is also protected as potential probing target. The length of the comparator is determined by the number of shield nets needed for the dedicated design. Both inputs of the comparator are connected to the same source nets, but one is the exact shield net from upper layer, while the other one is the copy form lower layer. These will be implemented in the routing constraint step as illustrated in Section III-E. Selection of the best layer for single layer shield and the best two layers for multilayer shield will be discussed in Section III-C. Next, floor-planning and wire routing of the design are constrained to build the internal shield and provide protection on target nets against probing attacks. Details of these procedures are given in Sections III-D and III-E. At last, to evaluate the protection performance of shielded designs, the EA metric [13] with additional realistic optimization is used as discussed in Section III-F.

A. Target Net Identification

In this section, we discuss how we identify the nets which are most likely to be targeted for probing. Nets that are connected to assets are the most likely to be probed. In addition, an attacker can also probe nets that are not directly connected to an asset, but still contain valuable information from which the asset can be derived. For example, let us assume that a two-input xor gate where one input is connected to an asset, e.g., encryption key, and the other input is connected to an input that an attacker can control, e.g., plain-text. Then the attacker can infer the asset by controlling the plain-text input to logic 0 and probing the output of the XOR gate because the asset input is consistent with the output when the other input of the XOR gate is logic 0. Therefore, in addition to nets that are directly connected to assets, other nets which can be exploited to extract the asset also need to be protected against probing attack. Since it is inefficient to protect all nets in an system-on-chip (SoC), we develop a probing target identification metric to rank the nets according to their ability to leak asset information and therefore, the nets' likelihood of being targeted for probing can be deduced. Note that this target identification metric only applies for the possible information leakage (IL) from pure signal propagation and simple logic combinations. Those nets that can be used to derive asset information by complicated mathematics process, e.g., the nets in the last round of an encryption module for typical fault injection attacks are not covered in the target identification metric. To protect this type of nets against probing attack, these nets can be declared as a kind of special assets in the user input.

Our anti-probing design flow first requires the designers to input the name of nets/ports where the asset is located, e.g., the name of key nets, as user input. Then our flow performs the target net identification technique to identify all nets which are likely to be targeted for probing attack. This technique utilizes a *target score* ($f_{\mathrm{TS}}(i)$) metric to quantify the likelihood of a net to be targeted in a probing attack. A higher value of *target score* indicates the net is more likely to be targeted in a probing attack. The attacker will prefer to probe those high *target score* nets because he/she can reveal more information with less effort on controlling signals from the net with higher *target score*. For each net i in the circuit

$$f_{\mathrm{TS}}(i) = \frac{f_{\mathrm{IL}}(i)}{f_{\mathrm{PD}}(i)+1} \tag{1}$$

where $f_{\mathrm{IL}}(i)$ denotes IL and quantifies the amount of asset information leaked by observing net i, and $f_{\mathrm{PD}}(i)$ indicates the difficulty to control the logic values of internal nodes to avoid the asset signal being muted and propagate it to net i based on the SCOAP controllability metric [15]. The $+1$ is to avoid 0 value at the denominator. A larger value of $f_{\mathrm{IL}}(i)$ means more asset information can be leaked at net i. On the other hand, a larger $f_{\mathrm{PD}}(i)$ value indicates that it is more difficult to propagate an asset signal to net i. Hence, a higher $f_{\mathrm{TS}}(i)$ represents a higher likelihood of being targeted for probing.

$f^{IL}(i)$ Calculation: IL of a net i quantifies how much sensitive information can be directly inferred if this net is probed and observed by the attacker. If k (0 or 1) is observed at net i, the $f_{\mathrm{IL}.k}(i)$ is defined as the number of asset bits that net i is associated with divided by the number of possible logic combinations of the associated asset bits to output k at net i. The overall IL $f_{\mathrm{IL}}(i)$ is the weighted summation of $f_{\mathrm{IL}.0}(i)$ and $f_{\mathrm{IL}.1}(i)$ based on the probability of observing 0 and 1 at net i. $f_{\mathrm{IL}}(i)$ is calculated for each net and is evaluated on a gate-by-gate basis from input to output. We use a two-input AND gate, as shown in Fig. 4(a), as an example to present how $f_{\mathrm{IL}}(i)$ is derived. Note that a similar process is used to evaluate $f_{\mathrm{IL}}(i)$ for all types of standard cell gates. We classify the IL calculation into the following three categories.

Case 1 (All Inputs Are Fanout Nets of Assets): In this case, all inputs of the gate are associated with the assets. Fig. 4(b) shows an example of case 1, where $a0$ and $a1$ are both asset signals. If an attacker probes the net $Z0$, then he/she can extract some information about the asset $a0$ and $a1$. We can use the following four equations to calculate the IL at $Z0$ ($f_{\mathrm{IL}}(Z0)$):

$$f_{C,k}(Z0) = \sum_{\mathrm{Gate}(m,n)=k} f_{C,m}(a0) \times f_{C,n}(a1) \times (k,m,n \in \{0,1\}) \tag{2a}$$

$$f_B(Z0) = f_B(a0) + f_B(a1) \tag{2b}$$

$$f_{\mathrm{IL},k}(Z0) = \frac{f_B(Z0)}{f_{C,k}(Z0)} \tag{2c}$$

$$f_{\mathrm{IL}}(Z0) = \sum_{k=0}^{1} f_{\mathrm{IL},k}(Z0) \times \frac{f_{C,k}(Z0)}{2^{f_B(Z0)}} = \frac{f_B(Z0)}{2^{f_B(Z0)-1}} \tag{2d}$$

where k, m, and n is the logic value: 0 or 1; $\mathrm{Gate}(m,n)=k$ is the gate function to make k at the output with two inputs m and n [m AND $n = k$, in Fig. 4(b) example]. Six numerical measures ($k = 0$ or 1) for input nets, e.g., $a0$ and $a1$, are considered as shown in Table I. All measures for asset nets (e.g., $a0$–$a2$) would be 1, while they would be 0 for nonasset nets that lie outside of any asset propagation path (e.g., $n0$–$n2$). These measures for nets in Fig. 4(b) are shown in Table II. The IL calculation for other types of gates is similar to AND gates. For all types of gates, (2a)–(2d) are the

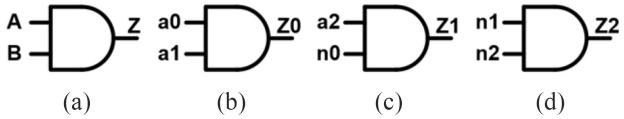


Fig. 4. AND gate examples. a0, a1, and a2 are asset signals. n0, n1, and n2 are non-asset signals.

TABLE I
MEASURES TO CALCULATE IL

Measures	Description
$f_{\mathrm{C,k}}(i)$	Number of asset signal combinations to output k (0 or 1) at net i
$f_B(i)$	Number of asset bits in the fan-in of net i
$f_{\mathrm{IL,k}}(i)$	Information leakage when net i is k (0 or 1)
$f_{\mathrm{IL}}(i)$	Overall information leakage of net i

TABLE II
IL MEASURES FOR NETS IN FIG. 4

Measures	a0	a1	Z0	a2	n0	Z1	n1	n2	Z2
$f_{C.0}(i)$	1	1	1+1+1=3	1	0	1	0	0	0
$f_{C.1}(i)$	1	1	1	1	0	1	0	0	0
$f_B(i)$	1	1	1+1=2	1	0	1	0	0	0
$f_{IL.0}(i)$	1	1	2/3	1	0	1	0	0	0
$f_{IL.1}(i)$	1	1	2/1	1	0	1	0	0	0
$f_{IL}(i)$	1	1	1/2+1/2=1	1	0	1	0	0	0

same, while the Gate function needs to update accordingly. However, the total $f_{\mathrm{IL}}(Z0)$ (2d) is not a function of $f_{C.0}(Z0)$ and $f_{C.1}(Z0)$, which means the total IL calculation for different types of gates is a uniform function of the number of asset bits that the calculated net is associated with. Therefore, only (2d) is needed to calculate the total IL for any net in the circuit. If we want to know the specific IL when a specific value, 0 or 1, is observed at net *i*, all four equations (2a)–(2d) should be calculated.

Case 2 (One of the Inputs Is Fanout Net of Assets): In this case, one input of the gate is associated with assets while the rest input is controllable by the attacker. Fig. 4(c) shows an example of case 2, where *a*2 is an asset net and *n0* is a nonasset net that is not associated with any asset but can be controlled by an attacker. Here, the attacker can control *n*0 to observe *a*2 from *Z*1. Therefore, the IL for *Z*1 is the same as asset input *a*2. The IL measures for nets in Fig. 4(c) are shown in Table II.

Case 3 (No Input Is Fanout Net of Assets): In this case, both inputs of the gate are nonasset signals that are not associated with any asset. Fig. 4(d) shows an example of case 3, where *n*1 and *n*2 are nonasset nets. Therefore, the IL for *Z*2 is 0. The IL measures for nets in Fig. 4(d) are shown in Table II.

Case 4 (One of the Inputs Is a Noncontrollable Constant Value): In this case, one input of the gate is associated with assets while the rest input is a constant value which cannot be controlled by attackers. There are two scenarios that may happen. One is that the constant value will propagate the asset signal to the output of the gate, e.g., if the constant value is 1 for an AND gate. In this case, the IL measures in Table I of the gate output would be the same with the asset input. The other situation is that the constant value will mute the asset signal, e.g., if the constant value is 0 for an AND gate. Then, the IL value of the gate output would be the same with a nonasset signal which is 0.

TABLE III
TARGET SCORE CALCULATION FOR NETS IN FIG. 4

Measures	a0	a1	Z0	a2	n0	Z1	n1	n2	Z2
$f_{\mathrm{IL}}(i)$	1	1	1	1	0	1	0	0	0
$f_{\mathrm{PD}}(i)$	0	0	0	0	1	1	0	0	0
$f_{\mathrm{TS}}(i)$	1	1	1	1	0	1/2	0	0	0

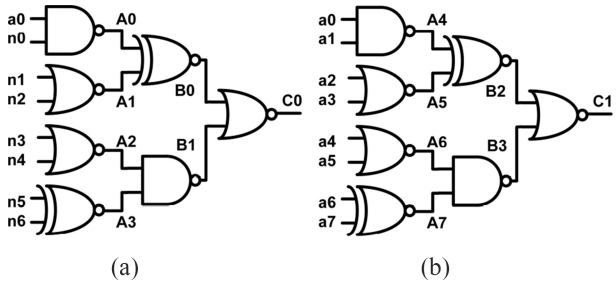


Fig. 5. Target score metric sample circuits. a0–a7 are asset signals. n0–n6 are non-asset signals.

TABLE IV
TARGET SCORE CALCULATION FOR NETS IN FIG. 5

Net	CC0	CC1	$\mathbf{f_{IL}}(i)$	$\mathbf{f_{PD}}(i)$	$\mathbf{f_{TS}}(i)$
a0-a7	INF	INF	1	0	1
n0-n6	1	1	0	0	0
A0	INF	2	1	$CC1_{\mathrm{n0}} = 1$	0.5
A1	2	3	0	0	0
A4-A7	INF	INF	1	0	1
B0	5	5	1	$CC1_{\mathrm{A1}} = 3$	0.25
B1	7	3	0	0	0
B2-B3	INF	INF	0.5	0	0.5
C0	4	9	1	$CC1_{\mathrm{A1}} + CC0_{\mathrm{B1}} = 10$	1/11
C1	INF	INF	1/16	0	1/16

$f_{PD}(i)$ Calculation: The $f_{\mathrm{PD}}(i)$ quantifies the difficulty to propagate asset information to net *i* using SCOAP combinational controllability metric (CC0 and CC1) [15]. When both inputs of a gate are fanout nets of asset which have nonzero IL value [e.g., Fig. 4(b)], there is no need to control other nets to propagate asset information to the output Z0. Therefore, the $f_{\mathrm{PD}}(Z0)$ for *Z*0 is set to 0. When one of the inputs is fanout net of asset [e.g., *a*2 in Fig. 4(c)], to propagate *a*2's information to *Z*1, *n*0 needs to be 1. $CC1_{n0}$ measures the 1-controllability value for net *n*0. Assuming *n*0 is a primary input, the $CC1_{n0}$ would be 1 and $f_{\mathrm{PD}}(Z1) = CC1_{n0} = 1$ for *Z*1. When net *i* is located *n* stages after asset signals, the $f_{\mathrm{PD}}(i)$ is the summation of *n* 1/0-controllability values of the nonasset input of the gate for each stage to propagate asset information to next stage.

Target Score Calculation: Table III shows the target score calculation using (1) for *Z*0–*Z*2 in Fig. 4(b)–(d), assuming *n*0–*n*2 are nonasset primary inputs. In Fig. 4(d), since both inputs are nonasset nets without any IL, the *target score* for *Z*2 is 0. Fig. 5 and Table IV show the target score metric calculation on two sample circuits where different types of gates and inputs are mixed. *a*0–*a*7 are asset signals while *n*0–*n*6 are nonasset primary inputs. In Fig. 5(a), the IL value ($f_{\mathrm{IL}}(i)$)

on *a0* propagation path ($a0$-$A0$-$B0$-$C0$) stays at 1, the target score is decreasing due to the difficulty to control nets ($n0 = 1$, $A1 = 1$, and $B1 = 0$) to propagate asset information to next stage increasing. On the other hand, in Fig. 5(b), all inputs are asset signals and the IL values decrease stage by stage, which indicates they are less and less likely to be targeted in probing attack.

To implement the target identification metric on a large circuit, the target score needs to be calculated from primary inputs to the primary outputs. A flip-flop can be treated as a buffer which maintains the target score and IL values as its input. For those feedback nets, in the initialization stage, they can be simplified as nonasset nets with zero target score and IL values. With the process of target score calculation, they will finally be assigned an updated value for target score and IL. Considering the sensitivity of the asset and the acceptable protection overhead, we can set a threshold value for target score to identify nets which are most likely targeted for probing attack. Any net whose target score is larger than this threshold value needs to be protected accordingly against probing attack. It can be observed from Fig. 5 and Table IV that the target scores for $\{C0, C1\}$ are much lower than the other nets closer to asset nets. Therefore, we may exclude $\{C0, C1\}$ from our target nets list, which indicates that only two level nets after asset need to be protected against probing attack. By further study, the target score threshold could be set to 0.125 to guarantee the two level nets after assets are protected. So, we would recommend the designers to set the target score threshold to at most 0.125 to achieve the default protection against probing attacks or lower to involve more target nets if the budget allows.

Note that the asset should be identified by the chip designer as a user input in the anti-probing design flow as shown in Fig. 3. If one of the assets is not identified in the user input, the target net identification metric would not be able to recognize the nets that can leak information of the unidentified asset.

This article is mainly focusing on the protection against probing attack. So, the target nets identification metric we proposed here is mainly developed to identify target nets in a probing attack. As a probing attacker, he/she tends to directly read out the critical information from the probed nets, which is the preferred nature of a probing attack without additional complicated analysis used in SAT attack or differential fault analysis (DFA). A universal target identification metric counting for a variety of attacks is out of the scope of this article because the principle to identify the target nets for various attacks differs a lot. Therefore, our metric is not intelligent enough to consider all types of attacks. For those nets that might be utilized to infer asset information through complicated mathematical analysis, e.g., the intermediate nets of an encryption/decryption process used in DFA technique, they are not covered by the target net identification metric. However, these nets can also be protected in our methodology by declaring them in the user input as part of "asset" or setting the target score threshold to 0 to protect all fan-out nets of the keys against probing attack.

B. Shield Net Identification

One unique feature that distinguishes our proposed anti-probing physical design flow from previously proposed techniques is the adoption of internal functional nets of the design as shield to protect target nets against probing attack. Existing active shield countermeasures are vulnerable to bypass attacks and reroute attacks [9], [11], [12], [18], [19] because the shield at the top-most layer is relatively easy to access and manipulate. In addition, more advanced active shields require cryptographically secure pattern generators [5], which themselves are sources of vulnerability and additional overhead. In contrast, utilizing internal functional nets provides the following major advantages. First, they will be routed within internal layers of the chip and therefore far more difficult to bypass and reroute. Second, the design itself will generate these signals alleviating the need for pattern generation, which will save the major area overhead introduced by active shield pattern generation. In this design, we develop a technique for identifying which internal nets can be utilized as shielding nets (covering nets). We define the following five requirements along with associated metrics as follows.

1) *Target Score:* The shield nets should not carry any asset information since they are not protected and could be probed. The prior target score approach can be inverted to identify nets that carry the least sensitive information.
2) *Toggle Frequency:* The shield nets should have a relatively high toggling rate so that an attacker cannot replace them with a constant value after cutting them.
3) *Switching Probability:* It should be difficult to predict the signals on shield nets, which requires the probability of the net being 1 or 0 to be balanced.
4) *Controllability:* The attacker should not have control over the shield nets. Otherwise, the shield can be replicated with the controlled value, allowing the attacker to freely perform the attack. The SCOAP controllability value [15] can be used for this feature and should be as high as possible.
5) *Delay Slack:* Using internal nets as shield nets should not impact the critical path delay and the design's performance. As discussed in the later sections, shield nets will need to be extended and moved to cover target nets in the design, which will hurt the timing of the paths that the shield net belongs to. Hence, they should not lie on critical paths.

For each of the aforementioned shield requirements, a threshold value of corresponding metric should be determined to maximize the coverage on target nets and minimize the vulnerabilities and impacts from shield nets. The final shield candidate nets will be the intersection of the five net collections which satisfy the threshold values for each shield requirement.

C. Best Shield Layer

After appropriate shield nets are identified, the metal layer in the chip layout to route these shielding nets needs to be determined. In this article, we build two types of shield structures: 1) single layer shield and 2) two layer parallel shield.

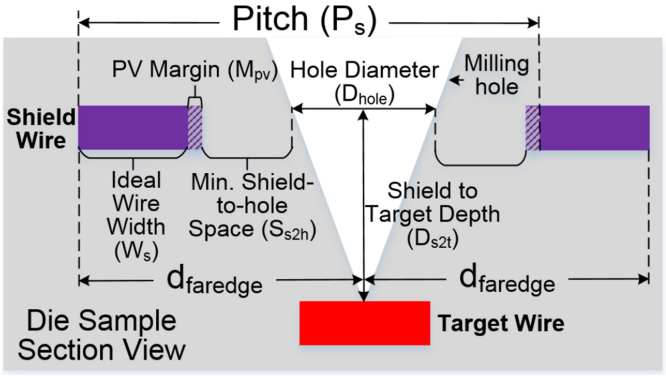


Fig. 6. Calculations for shield security and d_{faredge}.

TABLE V
SHIELD SECURITY IN SAED 32-nm LIBRARY

Max R_{FIB}	Shield Layer							
Target Layer	9	8	7	6	5	4	3	2
8	0.46	N/A						
7	0.86	0.64	N/A					
6	1.26	1.28	0.64	N/A				
5	1.66	1.91	1.28	1.81	N/A			
4	2.06	2.55	1.91	3.61	1.81	N/A		
3	2.46	3.19	2.55	5.42	3.61	4.41	N/A	
2	2.86	3.83	3.19	7.23	5.42	8.82	4.41	N/A
1	3.26	4.47	3.83	9.04	7.23	13.24	8.82	INF

For single layer shield, We assume a milling scenario using FIB technology as shown in Fig. 6, where colored bars are used to represent cross sections of metal wires on different routing layers. For the sake of argument, assume target wires (red) in the figure are on layer n, shield wires (purple) on layer $n+q$, and the attacker wishes to probe at one of the wires on target layer n to extract sensitive information. The hollowed-out cone (white) shown in the figure represents a cavity milled with FIB equipment. One known exploit on active shields is to create a reroute between identified equipotential points by circuit editing with FIB, so that the net would not become open when parts of the wires are removed [12]. This forces active shield designs to only use parallel wires with minimum spacing and widths [5] to maximize the attack complexity, because the shield with elbows (snake-like) may create a short path for reroute with a long section of the shield wire disabled.

From a layout point of view, active or analog shield designers are interested in the scenario where the attacker would make a mistake and leave a detectable footprint. To minimize the effect of the milling process, the attacker is likely to place the milling cavity in the center of two adjacent shield wires as shown in Fig. 6. To avoid affecting the normal signal transmission of shield wires, the attacker will avoid completely or partially cutting any shield wires. Further, a minimum space, S_{s2h}, is left between the shield wire and the milling cavity, as shown in Fig. 6, to minimize the effect of changed parasitic capacitance during the attack on the timing of shield wires. In order to account for the limitations of lithography and metalization as well, S_{s2h} is set to the same value with the minimum distance between metal wires as provided by the design rule of the technology. In addition, because of the process variation, the shield wires may be wider or thinner than the ideal wire width. Hence, to guarantee the minimum space between the shield wire and the milling cavity, an additional process variation margin (M_{pv}: typically 10% of the wire width) is added to the width of shield wire as shown in Fig. 6.

These restrictions create a maximal milling hole diameter limit on shield layer

$$D_{\text{hole}} < P_s - W_s - 2M_{pv} - 2S_{s2h} \tag{3}$$

where P_s is the pitch size of shield layer, W_s is the ideal width of shield wires, M_{pv} is the process variation margin of shield wires, and S_{margin} is the minimal space between the shield wire and the milling cavity which can be determined by the minimal space between metal wires defined by the technology design rule. The milling hole diameter is determined by

$$D_{\text{hole}} = \frac{D_{s2t}}{R_{\text{FIB}}} \tag{4}$$

where D_{s2t} is the depth from shield layer to target layer, and R_{FIB} is the aspect ratio of FIB, which is defined as the ratio between FIB depth D_{s2t} and diameter D_{hole} as shown in Fig. 6. Therefore, the maximum FIB aspect ratio that the shield could protect against, which is termed as shield security [19], can be modeled as

$$R_{\text{FIB,max}} = \frac{D_{s2t}}{P_s - W_s - 2M_{pv} - 2S_{s2h}}. \tag{5}$$

The higher the *shield security* ($R_{\text{FIB,max}}$) value is, the better the single layer shield is. The shield security can vary depending on shield layer, target layer, width of shield wire, and other layout technology parameters. Therefore, different technology libraries might lead to different shield security and different best shielding layer through (5). Table V shows the shield security calculated from SAED 32-nm library. As we can see, shield layer 6 has the best shield security for target nets on layers 3 and 4, and also good for target on layers 1 and 2. Though shield layer 4 is better than layer 6 for target nets on layers 1 and 2 in terms of shield security, it requires to route all target nets within only two layers (layers 1 and 2) to take this advantage, which may cause serious routing congestion. Hence, layer 6 is the overall optimal shield layer with excellent shield security and sufficient space left for routing of target nets for single layer shield designs. Therefore, in our single layer internal shield implementation, shield nets are routed on metal 6 and target nets are routed under metal 4 (metal 4 included). Compared to the conventional active shield approach whose shield wires are routed on the top-most layer (metal 9), the shield security for the best case of active shield (target on metal 1, shield on metal 9) is only 3.26, which is still less secure than the worst case of internal shield on $M6$ (target on metal 4, shield on metal 6) whose shield security is 3.61. In addition, the internal shield routed on metal 6 is more resistant to reroute attack where a shield path is duplicated between two equipotential points, and bypass attack where the shield is bypassed by leveraging the space between adjacent shield wires, since the wires beyond shield layer (layers 7–9) become huge obstacles to the attack.

Shield security is a very simple and useful metric to determine the best layer for single layer shield. However, it might

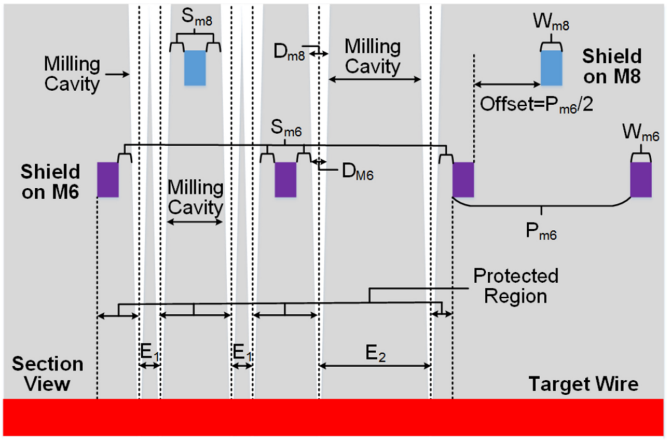


Fig. 7. Section view of a two layer staggered parallel shield.

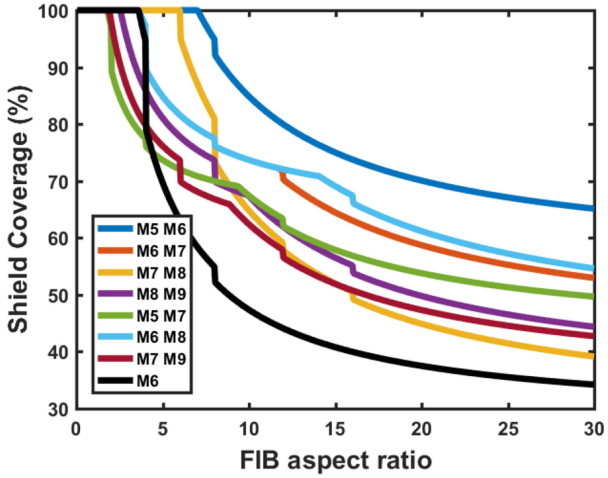


Fig. 8. Shield coverage of different shield designs in SAED 32-nm library.

not be appropriate for multilayer shield structures, e.g., two-layer parallel shield, because adding an extra layer of shield might not increase the maximum FIB aspect ratio that the shield can protect against resulting in the same shield security value. Though multilayer shield might improve the protected ratio against a specific FIB, as long as this ratio is not 100% the shield security will not be improved because it requires full protection. Therefore, to determine which layers are good for two-layer parallel shield, we propose the *shield coverage* metric.

Let us consider the two layer staggered parallel shield on *M*6 and *M*8 as shown in Fig. 7. The pitch size on *M*8 is twice of *M*6 as defined in SAED 32-nm library, and they have 50% offset to maximize the protection. The *shield coverage* is defined as

$$\text{Coverage} = \frac{\text{Protected Region}}{\text{Period}} = \frac{\text{Period} - \text{Exposed}}{\text{Period}}. \quad (6)$$

The *period* is the pitch size of the upper shield layer (P_{m8}) because typically the upper layer has a larger pitch size than the lower layer. The *exposed* is the region on target wires that is free to probe without triggering the shield alarm, which can be calculated as

$$\text{Exposed} = 2 \times E_1 + \left(\frac{P_{\text{upper}}}{P_{\text{lower}}} - 1\right) \times E_2 \quad (7)$$

where P_{upper} and P_{lower} are the pitch size of upper and lower shield layers (P_{m8} and P_{m6} in Fig. 7), and E_1 and E_2 are two types of exposed region as shown in Fig. 7) and are defined as

$$E_1 = \frac{1}{2}P_{\text{lower}} - \frac{1}{2}\left(W_{\text{upper}} + W_{\text{lower}}\right) - \left(S_{\text{lower}} + S_{\text{upper}}\right) - \frac{1}{2}\left(D_{\text{upper}} + D_{\text{lower}}\right) \quad (8a)$$

$$E_2 = P_{\text{lower}} - W_{\text{lower}} - 2S_{\text{lower}} - D_{\text{lower}} \quad (8b)$$

where W_{upper} and W_{lower} are the metal width of upper and lower shield layers (W_{m8} and W_{m6}), S_{upper} and S_{lower} are the space between shield wire and milling cavity (S_{m8} and S_{m6} which can be determined by the minimal metal space defined by the technology design rules), and D_{upper} and D_{lower} are the milling cavity diameter on the upper and lower shield layers (D_{m8} and D_{m6}) which can be calculated using (4).

The higher of the *shield coverage*, the better of the two-layer parallel shield design. As illustrated in (7) and (8), the *shield coverage* depends on many factors defined by the technology and the selection of shield and target layers. Fig. 8 shows the shield coverage of different two-layer staggered parallel shield designs using SAED 32-nm library. From the figure, we can see that all two-layer shield designs perform better than the single layer shield on *M*6 design (black curve) especially when R_{FIB} is high. Though the two-layer shield on *M*5 and *M*6 is theoretically optimal for shield coverage, it brings a practical issue, routing congestion, due to the small pitch size on *M*5. Therefore, the shield on *M*6 and *M*8 (second best in shield coverage) are implemented for evaluation in the later sections.

D. Floor-Planning Constraints

In conventional design flows, CAD tools perform floor-planning to optimize timing, power, and area. In an original design as shown in Fig. 9(a), target nets and the blocks containing them (red) are distributed randomly throughout the design. It is neither easy nor efficient to protect them with such placement. It might also require more shield nets than available. A more advantageous approach is to constrain them into a regularly shaped region, e.g., a rectangle, as shown in Fig. 9(b). This can be implemented by enumerating all gates connected to target nets, and then creating a floorplan group to constrain their relative placements. The location of this floorplan group is chosen to remain as close to its original placement to reduce the impact on performance. The optimal dimensions of this floorplan group are found by extracting all gates and nets involved into a sub-layout where only these gates and nets are placed and routed.

The comparator is used to detect the attack by comparing the shield signal from upper layer and another copy from lower layer. So the comparator nets should also be protected like target nets because if these nets are tampered to maintain a static value the testability of the shield nets will be compromised.

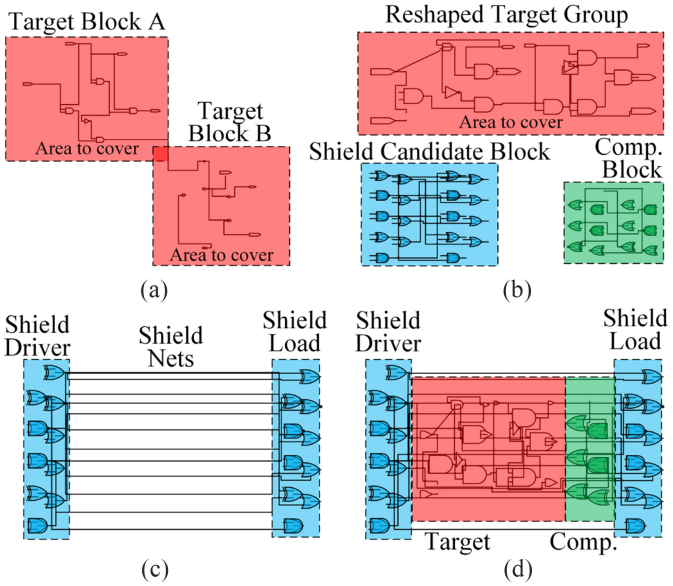


Fig. 9. (a) Irregular blocks (red) of sensitive target nets. (b) Reshape the sensitive target blocks to one regular rectangle block (red), shield candidate block (blue), and comparator block (green). (c) Shield gates (blue) are divided into shield nets driver block and shield nets load block. (d) Shield gates are placed surrounding the target and comparator blocks which will be covered by shield nets.

Hence, the comparator gates (green) are constrained in a floorplan group besides the target block as shown in Fig. 9(d). The comparator is XNOR based, which is designed to output 1 when there are no attacks, and optimized by design compiler for less area.

Unlike target nets, we divide the gates connected to shield nets into two separate floorplan groups: 1) shield nets driver group and 2) shield nets load group as shown in Fig. 9(c). Our proposed shield net identification metric ensures that the performance overhead due to our constrained floor-planning is minimal. Both shield nets driver group and load group (blue) are constrained at opposite ends of the expected shielding area (target and comparator block) as shown in Fig. 9(d), so that routing of shield nets crosses the target area and therefore provides vertical protection from milling/probing. The shield nets load group should be placed at the comparator's side. So that the received signals from shield nets could be compared in the comparator. The driver and load gates of the shield nets are buffers. Larger buffers are used for drivers to provide better driving capability for the long shield nets.

E. Routing Constraints

In addition to creating floor-planning constraints, wire-routing constraints are also necessary to protect the device against probing attacks with large aspect ratio FIB. Aspect ratio of FIB is defined as the ratio between depth D and diameter d, as shown in Fig. 10, of a milled cavity and is an important measure of FIB performance [13]. A larger aspect ratio results in a milling cavity of smaller diameter on the top-most exposed layers, and therefore has less impact on the protective circuitry. Section III-C has revealed the best

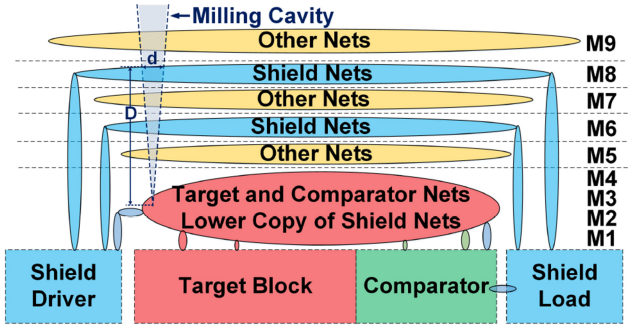


Fig. 10. Routing layer constraints for target and shield nets.

shield layers for single layer shield design and two-layer parallel shield design to maximize the protection against probing attack. Further, routing target nets in lower layer can also increase the chance to be protected from other nonshield internal function nets in the design. In this article, we route shield nets on $M6$ ($M9$ is the top layer) for single layer shield design, route shield nets on $M6$ and $M8$ for two-layer parallel shield design, and route target nets and comparator nets under $M4$ ($M4$ included) to get an optimal protection as shown in Fig. 10. Further, another copy of the shield nets, which is the other branch of the driving buffers of the shield nets, are also routed under $M4$ to be compared with the genuine shield nets on the upper layers at the comparator.

F. Exposed Area Calculation

To assess the design's vulnerability to probing attacks, Shi *et al.* [13] proposed an EA metric by assuming that a complete cut of one shield wire is required for the detection of the attack. However, it is too conservative in several aspects. The first is assuming only complete cuts will be sufficient for detection. In reality, as soon as a minimum cross section of a cut wire to ensure correct signal transmission on the shield wires is violated, the attack is likely detected by active shield. Further, even if the milling cavity does not touch the shield wires, the changed parasitic capacitance due to the close distance between the shield wires and the milling cavity could possibly result in significant delays on shield wires especially in the low power designs, and thus trigger the alarm of an active shield by violating the setup time of flip flops. In addition, the attackers will probably not try to challenge the sensitivity of the shield. They will always set a margin for the potential timing violation, and also the FIB errors or operation mistakes. So, a space margin between the shield and the milling hole is necessary for attackers to improve the attack success rate. A more realistic model for detection is shown in Fig. 6, i.e., the probing attack can be detected if the center of milling exists within d_{faredge} from the far edge of the shield wire, where

$$d_{\text{faredge}} = \frac{D_{s2t}}{2R_{\text{FIB}}} + W_s + S_{s2h} + M_{pv} \tag{9}$$

where D_{s2t}, R_{FIB}, W_s, S_{margin}, and M_{pv} are defined in Section III-C. Equation (9) shows the possibility to find the area which milling center should not fall inside. We term this

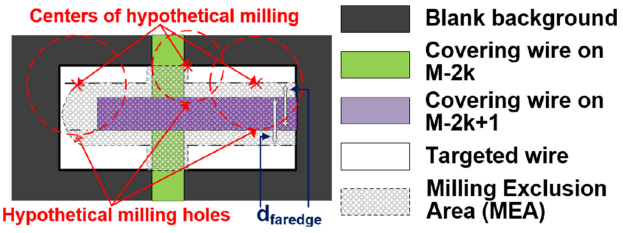


Fig. 11. EA calculation.

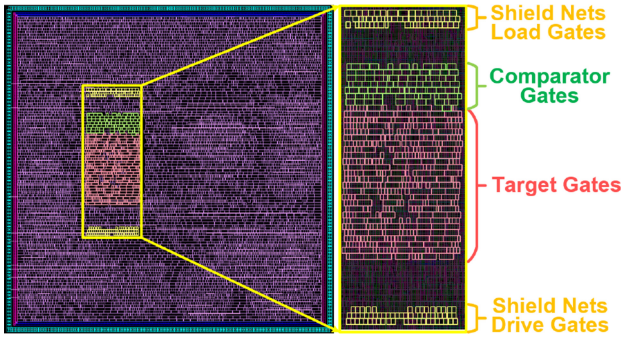


Fig. 12. Grouped and reshaped target gates, comparator gates, and shield gates in AES.

area the milling exclusion area (MEA). The desired EA will be its complement projected on the target layer.

Fig. 11 shows how the EA can be found for any given target wire and covering wires on higher layers which are capable of projecting the MEA. Assuming the white region is the targeted wire at lower layer of a layout and the green and purple regions are the covering wires at upper layers above the targeted wire, the shaded region is the MEA, which indicates that if the milling center falls in this area then the attack will be detected. Hence, the complement area of MEA is the desired EA that will not cause any risk to be detected. The EA can vary according to the different aspect ratio of FIB, since the diameter of the cavities milled by FIB with different aspect ratio is different. Larger EA in the design represents more vulnerable to probing attacks.

IV. Evaluation

In this section, the proposed FIB-aware anti-probing physical design flow is evaluated to find out how efficient the design flow can be and how much area in the design is vulnerable to probing attacks. For this purpose, layout of AES and DES crypto-cores are selected for the evaluation of the proposed design flow.

A. Implementation of Proposed Design Flow

The DES and AES modules used are from OpenCores [14]. They are described in register-transfer level (RTL) code and synthesized using Synopsys *Design Compiler* with Synopsys SAED 32-nm technology library. The layout of AES and DES modules are generated and constrained using Synopsys *IC Compiler*. The asset in the AES and DES modules is taken to be the encryption key (128 bits for AES and 56 bits for DES), which is hardcoded in the design.

TABLE VI
Threshold Values for Shield Nets Identification in AES

Metric	Min.	Max.	Best	Percentage	Threshold
target score	0	1	0	80%	<0.001
Togg. Rate	0	0.06	0.06	40%	>0.0187
Delay Slack	0.01	1.60	0.01	40%	<1.23
CC0(SCOAP)	0	2532	2532	40%	>395
CC1(SCOAP)	0	2081	2081	40%	>332
Probability	0	1	0.5	40%	0.22∼0.78

The *target score* metric illustrated in Section III-A is used to identify the probing target nets in the AES and DES modules. The *target score* threshold value is set to 0.125 (target score for asset net is 1, for nonasset net is 0), which results in nets within two levels after the asset nets being identified as probing target nets. Hence, 384 nets for AES and 200 nets for DES, including key nets are probing target nets in the two designs. Further, gates connected to target nets are grouped and reshaped into a rectangular target block as shown in Fig. 12 (red). In addition, a 64-bit comparator is inserted in the AES and DES designs. Comparator gates are also grouped and reshaped into a rectangular block besides target gates block as shown in Fig. 12 (green).

Table VI shows the metrics and threshold values used to identify shield nets in AES module to cover target block. The *Min.* and *Max.* columns shows the minimum and maximum value measured in the design for each metric. The *Best* column indicates the optimal value as shield net for each metric. The optimal value for the metric of shield nets are the minimum values of target score and delay slack; and maximum values of Togg. Rate, CC0, and CC1. The *Percentage* column presents percentage of all nets that are picked for each metric. The *Threshold* column indicates the threshold values for each metric, which are determined to offer a balanced trade-off between security and overhead. Hence, 136 nets in AES module and 118 nets in DES module, which meet all requirements of shield metrics, are identified as shield candidate nets for both designs. The final number of shield nets used for building the internal shield depends on the area on chip that needs to be protected against probing attack and the structure of the shield (single layer or two-layer). In our implementation, 64 and 56 shield nets are used to build the single layer internal shield for AES and DES, respectively. Therefore, in AES module, 64 driver gates and 64 load gates connected to the shield nets are reshaped into two groups, respectively, and placed at the opposite ends of target and comparator block as shown in Fig. 12 (yellow). Fig. 13 shows the routing of target nets [Fig. 13(a)], shield nets [Fig. 13(b)], and their layer distribution [Fig. 13(c) and (d)] in the AES layout. Target nets, comparator nets, and shield nets copy are constrained in the reshaped target and comparator block and routed under $M4$ as discussed in Section III-E. Most shield nets are routed on $M6$ to provide optimal coverage.

In addition to the single layer internal shield designs, two-layer staggered parallel shield [18], which utilizes two routing layers to build the parallel shield with some offset between different layers, can provide better protection. Fig. 14(a) shows an example of the two-layer staggered shield on $M6$ and $M8$.

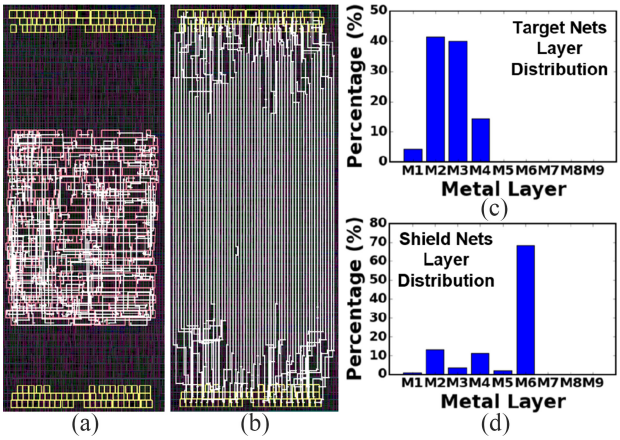


Fig. 13. (a) AES shield gates (yellow), target gates (red), and highlighted target nets under $M4$ (white). (b) AES shield gates (yellow) and highlighted shield nets on $M6$ (white). (c) Target nets layer distribution. (d) Shield nets layer distribution.

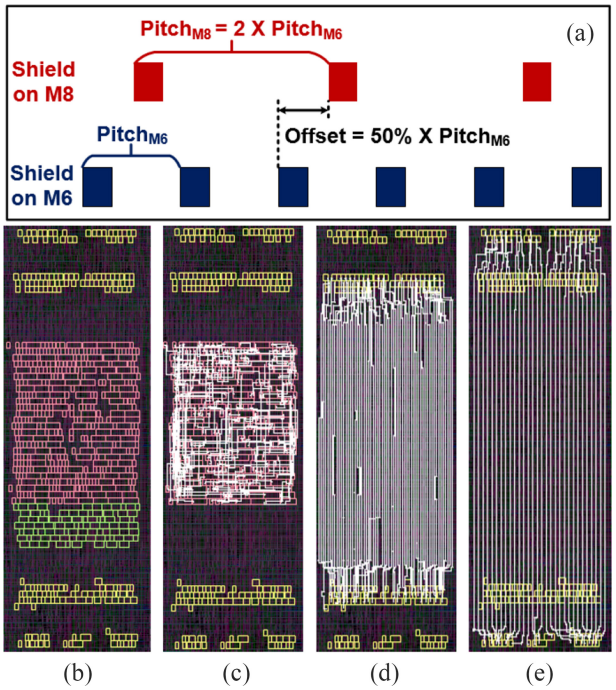


Fig. 14. (a) Diagram of two-layer staggered shield on $M6$ and $M8$. (b) Placement of target gates (red), comparator gates (green), and shield gates (yellow). Routing of (c) target nets, (d) shield nets on $M6$, and (e) shield nets on $M8$.

The pitch size on $M8$ is 2 times of the pitch size on $M6$ in SAED 32-nm library, which results in that the shield density on $M8$ being half of the shield density on $M6$. The 50% offset is set between the shield wires on $M6$ and $M8$ to maximize the protection. Fig. 14(b) shows the placement of target gates (red), comparator gates (green), and shield gates (yellow). Fig. 14(c)–(e) shows the routing of target nets, shield nets on $M6$, and shield nets on $M8$, respectively.

Besides the baseline single layer shield design and two-layer parallel shield design as illustrated above, we also implement four extra different designs for AES and DES, respectively, to show the high efficiency of our anti-probing physical design flow. Table VII shows the description of implemented six different designs for AES and DES. Design no. 1 is the original design using conventional place and route flow without any protection against probing attack. Design no. 2 is the baseline single layer shield on $M6$ as illustrated before. Design no. 3 decreases the target score threshold from 0.125 to 0.01, which involves more target nets protected under the internal shield. Design no. 4 includes those common fault injection target nets in the asset declaration, so that the nets vulnerable to fault injection attack are also protected under the shield. Design no. 5 is the two-layer staggered parallel shield as shown in Fig. 14. Design no. 6 is the conventional active shield design with a lightweight Simon cipher inserted as the shield signal pattern generator [5]. Fig. 15 shows the diagram of a conventional active shield [Fig. 15(a)] and the layout of the implemented active shield on AES [Fig. 15(b)]. The number of total gates, target nets, and target gates for different designs are listed in the third, fourth, and fifth columns of Table VIII.

Table VIII also shows the timing, power, area and routing overhead of all these designs compared to the original AES

TABLE VII
DESCRIPTION OF IMPLEMENTED DESIGNS FOR AES AND DES

No.	Design	Notes
1	Original Design	Conventional physical design flow
2	Single layer Shield I	Single layer shield on M6
3	Single layer Shield II	Decrease target score threshold to 0.01 to include more target nets
4	Single Layer Shield III	Include nets of fault injection positions in the Asset
5	Two-layer Parallel Shield	Two-layer shield on M6 and M8
6	Active Shield	Conventional active shield design

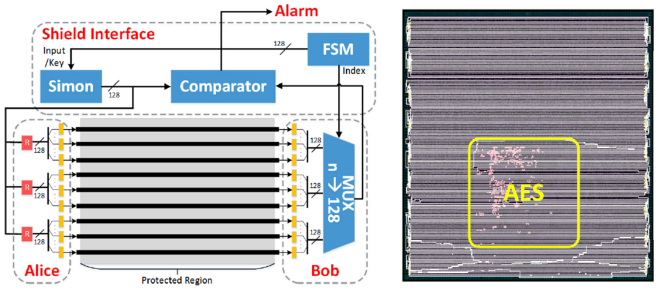


Fig. 15. (a) Diagram of a conventional active shield. (b) Layout of the implemented conventional active shield on AES.

TABLE VIII
CONFIGURATION AND OVERHEAD OF DIFFERENT AES AND DES DESIGNS

Module	Design	Total Gates	Target Nets	Target Gates	Timing	Power	Area	Routing
AES	2	10293	384	512	0.32%	2.79%	0.74%	11.60%
AES	3	10293	480	672	0.66%	3.66%	3.02%	14.80%
AES	4	10293	535	833	0.66%	6.03%	3.17%	22.99%
AES	5	10293	384	512	0.34%	4.90%	1.44%	17.77%
AES	6	28048	384	512	3.95%	439.83%	402.31%	407.40%
DES	2	8882	200	432	1.18%	0.75%	0.51%	10.39%
DES	3	8882	376	637	4.55%	1.38%	0.50%	13.41%
DES	4	8882	481	783	4.55%	1.67%	0.80%	21.16%
DES	5	8882	200	432	1.18%	2.83%	1.78%	20.85%
DES	6	26637	200	432	3.64%	365.17%	413.91%	556.54%

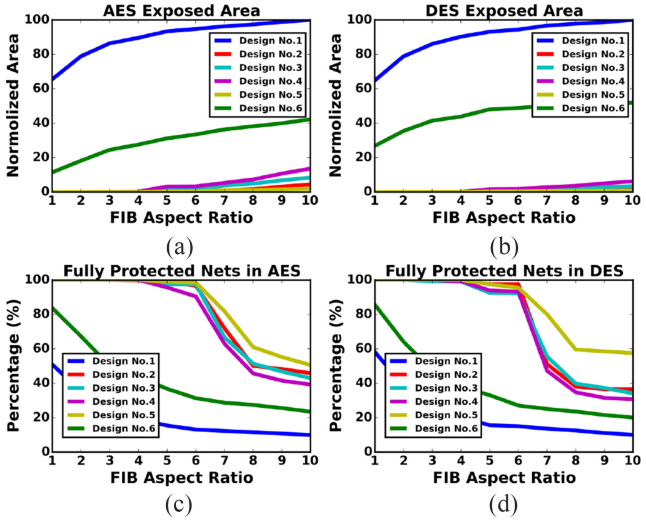


Fig. 16. EA in (a) AES and (b) DES. Percentage of fully protected target nets in (c) AES and (d) DES.

and DES without any constraints. Note that design nos. 2 and 5 have the same number of total gates, target nets, and target gates, but they are implemented with different shield structure as shown in Table VII. As a result, their overhead is different. As we can see from the table, the overhead of the baseline single layer shield (design no. 2) is less than 3% for both AES and DES in timing, power, and area. In addition, the timing, power, and area overhead of all internal shield approaches (design nos. 2–5) are all less than 6% even with lower target threshold (design no. 3) or including fault injection target nets (design no. 4), which indicates that even if we increase our security standard to protect more sensitive nets against probing attack, the overhead is still acceptable and not increased too much. Further, if considering the overhead to an SoC, this amount of overhead could be completely ignored since AES or DES module is just a very small portion in an SoC. However, the conventional active shield approaches (design no. 6) have ∼400% overhead in power, area, and routing, which is much larger than our internal shield designs, because it requires the insertion of shield signal pattern generator and other supporting circuitry, e.g., FSM.

B. Exposed Area

The proposed internal shielding approach against probing attack is evaluated by the EA metric illustrated in Section III-F. Fig. 16(a) and (b) shows the normalized EA of all types of designs in Table VII for AES [Fig. 16(a)] and DES [Fig. 16(b)]. The EA is calculated across FIB aspect ratio from 1 to 10. As the FIB aspect ratio increases, the EA for all designs will also increase since d_{faredge} decreases with larger FIB aspect ratio as shown in (9), which results in smaller MEA and thus larger EA. By using the proposed anti-probing design flow, the EA of all internal shield designs (design nos. 2–5) can be reduced to 0 for both AES and DES when the FIB aspect ratio is low. Even with the advanced FIB (aspect ratio is 10), the EA of baseline single layer shield (design no. 2) and two-layer shield (design no. 5) can be reduced at least to 5% and 2%, respectively, for both AES and DES. Fig. 16(c) and (d) shows the percentage of fully protected target nets for all designs. We define a net that is fully protected as one that does not have any EA. From Fig. 16(c) and (d), almost 100% of target nets for all internal shield designs (design nos. 2–5) are fully protected when $R_{\text{FIB}} \leq 6$, while less than 20% of target nets are fully protected for original AES and DES designs when $R_{\text{FIB}} = 6$. With the advanced FIB, there are still 50% and 60% of target nets fully protected under two-layer staggered shield (design no. 5) for AES and DES, respectively, which is about five times more than the original AES and DES designs. For design nos. 2–4 with same single layer shield protection but different target nets configuration and increasing overhead as shown in Table VIII, Fig. 16(c) and (d) shows that they have similar security performance which indicates that our internal shield design flow can provide guaranteed protection with different target nets configuration. Compared with internal shield designs, conventional active shield designs (design no. 6) can only reduce the EA to ∼40% and increase the number of fully protected nets by about two times, which is not efficient as shown in Fig. 16(a)–(d).

V. Conclusion

In this article, we presented the FIB-aware anti-probing physical design flow, which incorporates three security-critical steps in the conventional physical design flow. The floorplanning and routing of the design are constrained to provide coverage on asset nets through internal shield. Evaluations on AES and DES modules show that the total vulnerable EA to probing attack of the anti-probing design can be decreased by 100% compared to original design with all target nets fully protected, while the performance of the conventional active shield approach is inefficient. In addition, the general overhead of all internal shield designs are less than 6%, which can be totally ignored in an SoC.

Moreover, due to the unavoidable and stochastic wire shift, process variation, and probe tip shift, the effective probing area will shrink further, which makes the probe-able target area on the chip even more limited. When FIB aspect ratio is high, even if all probe-able asset nets could be compromised, there are still considerable asset nets (>50%) that are fully protected. For long-bit assets like keys, extracting the remaining asset information is still exponentially difficult. In the future work, we will apply our anti-probing design flow to SoCs which contain more types of asset needing to be protected and propose effective countermeasure against back-side probing attack.

References

[1] S. Skorobogatov, "Physical attacks on tamper resistance: Progress and lessons," in *Proc. 2nd ARO Spectr. Workshop Hardw. Assurance*, Washington, DC, USA, 2011, pp. 1–16.

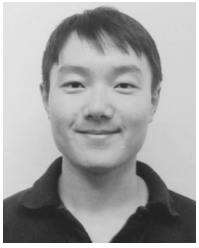
[2] S. E. Quadir *et al.*, "A survey on chip to system reverse engineering," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 33, no. 1, 2016, Art. no. 6.

[3] V. Sidorkin, E. Veldhoven, E. V. D. Drift, P. Alkemade, H. Salemink, and D. Maas, "Sub-10-nm nanolithography with a scanning helium beam," *J. Vacuum Sci. Technol. B*, vol. 27, no. 4, pp. 18–20, 2009.

[4] H. Wu *et al.*, "Focused helium ion beam deposited low resistivity cobalt metal lines with 10 nm resolution: Implications for advanced circuit editing," *J. Mater. Sci. Mater. Electron.* vol. 25, no. 2, pp. 587–595, 2014.

[5] J.-M. Cioranesco *et al.*, "Cryptographically secure shields," in *Proc. IEEE Int. Symp. Hardw. Oriented Security Trust (HOST)*, May 2014, pp. 25–31.

[6] M. Ling, L. Wu, X. Li, X. Zhang, J. Hou, and Y. Wang, "Design of monitor and protect circuits against FIB attack on chip security," in in *Proc. 8th Int. Conf. Comput. Intell. Security (CIS)*, Nov. 2012, pp. 530–533.
[7] M. Weiner, S. Manich, R. Rodríguez-Montañés, and G. Sigl, "The low area probing detector as a countermeasure against invasive attacks," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 2, pp. 392–403, Feb. 2018.
[8] Y. Ishai, A. Sahai, and D. Wagner, "Private circuits: Securing hardware against probing attacks," in *Advances in Cryptology—CRYPTO*, D. Boneh, Ed. Berlin, Germany: Springer, 2003, pp. 463–481.
[9] H. Wang, D. Forte, M. Tehranipoor, and Q. Shi, "Probing attacks on integrated circuits: Challenges and research opportunities," *IEEE Des. Test.*, vol. 34, no. 5, pp. 63–71, Oct. 2017.
[10] *Building a Secure System Using TrustZone Technology*, ARM Inc., Cambridge, U.K., Accessed. Jul. 13, 2017.
[11] V. Ray, "FREUD applications of FIB: Invasive FIB attacks and countermeasures in hardware security devices," in *Proc. East Coast Focused Ion Beam User Group Meeting*, Feb. 2009, pp. 2–17.
[12] C. Tarnovsky. (2013). *Deconstructing a 'Secure' Processor*. Accessed: Nov. 14, 2019. [Online]. Available: https://www.youtube.com/watch?v=w7PT0nrK2BE
[13] Q. Shi, N. Asadizanjani, D. Forte, and M. M. Tehranipoor, "A layout-driven framework to assess vulnerability of ICs to microprobing attacks," in *Proc. IEEE Int. Symp. Hardw. Oriented Security Trust*, McLean, VA, USA, May 2016, pp. 155–160.
[14] *AES (Rijndael) IP Core*. Accessed: Nov. 14, 2019. [Online]. Available: https://opencores.org/projects/aes_core
[15] L. H. Goldstein and E. L. Thigpen, "SCOAP: Sandia controllability/observability analysis program," in *Proc. 17th Design Autom. Conf.*, Minneapolis, MN, USA, Jun. 1980, pp. 190–196
[16] *Chipworks*. Accessed: Nov. 14, 2019. [Online]. Available: http://www.techinsights.com/
[17] R. Cole and J. Yakura, "Integrated circuit protection device and method," WO Patent WO1 997 036 326, Feb. 10, 1997.
[18] Q. Shi, H. Wang, N. Asadizanjani, M. M. Tehranipoor, and D. Forte, "A Comprehensive analysis on vulnerability of active shields to tilted microprobing attacks," in *Proc. Asian Hardw. Oriented Security Trust Symp. (AsianHOST)*, Hong Kong, 2018, pp. 98–103.
[19] H. Wang, Q. Shi, D. Forte, and M. M. Tehranipoor, "Probing assessment framework and evaluation of antiprobing solutions," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 6, pp. 1239–1252, Jun. 2019.
[20] *Pix2Net*. Accessed: Nov. 14, 2019. [Online]. Available: http://micronetsol.net/pix2net-software/
[21] *ChipJuice*. Accessed: Nov. 14, 2019. [Online]. Available: https://www.texplained.com/about-us/chipjuice-software/
[22] M. Weiner, W. Wieser, E. Lupon, G. Sigl, and S. Manich, "A calibratable detector for invasive attacks," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 5, pp. 1067–1079, May 2019.
[23] C. Helfmeier, C. Boit, and U. Kerst, "On charge sensors for FIB attack detection," in *Proc. IEEE Int. Symp. Hardw. Oriented Security Trust*, San Francisco, CA, USA, 2012, pp. 128–133.



Huanyu Wang (S'16) received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2014, and the M.S. degree in electrical engineering from Northwestern University, Evanston, IL, USA, in 2016. He is currently pursuing the Ph.D. degree in computer engineering with the Florida Institute for Cybersecurity, University of Florida, Gainesville, FL, USA.

His current research interests include hardware security and trust, very large-scale integration (VLSI) design, VLSI computer-aided design, and clam chowder recipe.

Qihang Shi (S'10–M'17) received the Doctorate degree in computer engineering from the University of Connecticut, Storrs, CT, USA, in 2017.

He is currently a Post-Doctoral Associate with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA. His research interests include hardware security and trust, and very large-scale integration test and reliability.

Adib Nahiyan (S'15) received the B.S. degree in electrical engineering from the Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2014. He is currently pursuing the Ph.D. degree with the Electrical and Computer Engineering Department, University of Florida, Gainesville, FL, USA, supported by the Semiconductor Research Corporation and Cisco.

He worked as a Research Intern with Cisco, Raleigh, NC, USA. His research interest includes hardware security, secure very large-scale integration design, and finding vulnerabilities in ASIC design.

Domenic Forte (S'09–M'13–SM'18) received the B.S. degree in electrical engineering from Manhattan College, Riverdale, NY, USA, in 2006, and the M.S. and Ph.D. degrees in electrical engineering from the University of Maryland, College Park, MD, USA, in 2010 and 2013, respectively.

From 2013 to 2015, he was an Assistant Professor with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT, USA. He is currently an Associate Professor with the Electrical and Computer Engineering Department, University of Florida, Gainesville, FL, USA, where he has been since July 2015. His research covers the entire domain of hardware security from nano devices to printed circuit boards, such as hardware security primitives, hardware Trojan detection and prevention, security of the electronics supply chain, security-aware design automation tools, reverse engineering, and anti-reverse engineering.

Dr. Forte was a recipient of the Presidential Early Career Award for Scientists and Engineers, the NSF Faculty Early Career Development Program Award, and the Army Research Office Young Investigator Award, and nine best paper awards and nominations. He serves on the organizing committees of top conferences in hardware security, such as the IEEE Symposium on Hardware Oriented Security and Trust (HOST) and AsianHOST. He also serves and has served on the technical program committees in the areas of electronic design automation, very large-scale integration design and test, and cybersecurity.

Mark M. Tehranipoor (S'02–M'04–SM'07–F'18) received the Ph.D. degree from the University of Texas at Dallas, Richardson, TX, USA, in 2004.

He is currently the Intel Charles E. Young Preeminence Endowed Chair Professor of Cybersecurity and the Founding Director of the Florida Institute for Cybersecurity, University of Florida (UF), Gainesville, FL, USA. Prior to joining UF, he served as the Founding Director of the CHASE and CSI Centers, University of Connecticut, Storrs, CT, USA. He has published over 400 journal articles and refereed conference papers and has given about 200 invited talks and keynote addresses. He has published 11 books and more than 20 book chapters. His current research projects include hardware security and trust, supply chain security, Internet of Things security, and very large-scale integration design, test, and reliability.

Dr. Tehranipoor was a recipient of a dozen best paper awards and nominations, as well as the 2008 IEEE Computer Society (CS) Meritorious Service Award, the 2012 IEEE CS Outstanding Contribution, the 2009 NSF CAREER Award, and the 2014 AFOSR MURI Award. He serves on the program committee of more than a dozen leading conferences and workshops. He has also served as the Program Chair of a number of IEEE and ACM sponsored conferences and workshops, such as Hardware-Oriented Security and Trust (HOST), ITC, DFT, D3T, DBT, and NATW. He has co-founded the IEEE International Symposium on HOST and served as the HOST 2008 and HOST 2009 General Chair. He is currently serving as a Founding Editor-in-Chief for the *Journal on Hardware and Systems Security*, and an Associate Editor for JETTA, JOLPE, the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, and ACM TODAES. He is a Golden Core Member of the IEEE CS, and a member of ACM and ACM SIGDA.