Is Split Manufacturing Secure?

Jeyavijayan (JV) Rajendran†, Ozgur Sinanoglu‡, and Ramesh Karri†
†Polytechnic Institute of New York University, ‡New York University – Abu Dhabi
Email: jrajen01@students.poly.edu, os22@nyu.edu, rkarri@poly.edu

***Abstract*— Split manufacturing of integrated circuits (IC) is being investigated as a way to simultaneously alleviate the cost of owning a trusted foundry and eliminate the security risks associated with outsourcing IC fabrication. In split manufacturing, a design house (with a low-end, in-house, trusted foundry) fabricates the Front End Of Line (FEOL) layers (transistors and lower metal layers) in advanced technology nodes at an untrusted high-end foundry. The Back End Of Line (BEOL) layers (higher metal layers) are then fabricated at the design house's trusted low-end foundry. Split manufacturing is considered secure (prevents reverse engineering and IC piracy) as it hides the BEOL connections from an attacker in the FEOL foundry. We show that an attacker in the FEOL foundry can exploit the heuristics used in typical floorplanning, placement, and routing tools to bypass the security afforded by straightforward split manufacturing. We developed an attack where an attacker in the FEOL foundry can connect 96% of the missing BEOL connections correctly. To overcome this security vulnerability in split manufacturing, we developed a fault analysis-based defense. This defense improves the security of split manufacturing by deceiving the FEOL attacker into making wrong connections.**

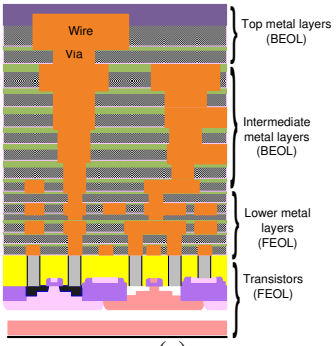
I. Introduction

Integration of digital, analog, radio frequency, photonic and other devices into a complex System on Chip (SoC) has been demonstrated [2]. More recently, sensors, actuators, and biochips are also being integrated into these already powerful SoCs. On one hand, SoC integration has been enabled by advances in mixed system integration and the increase in the wafer sizes (currently about 300 mm and projected to be 450mm by 2018 [2]). It has also reduced the cost per chip of such SOCs. On the other hand, support for multiple capabilities and mixed technologies has increased the cost of ownership of advanced foundries. For instance, the cost of owning a foundry will be $5 billion in 2015 [9]. Consequently, only large commercial foundries now manufacture such high performance, mixed system SoCs especially at the advanced technology nodes [10]. Absent the economies of scale, many of the design companies cannot afford owning and acquiring expensive foundries and hence outsource their design fabrication to these "one-stop-shop" foundries[1].

Globalization of Integrated Circuits (IC) design flow has led to several security vulnerabilities. If a design is fabricated in a foundry that is not under the direct control of the (fabless) design house, attacks such as reverse engineering, malicious circuit modification and Intellectual Property (IP) piracy are possible [10]. An attacker, anywhere in this design flow, can reverse engineer the functionality of an IC/IP, and then steal and claim ownership of the IP. An untrusted IC foundry may overbuild ICs and sell them illegally. Finally, rogue elements in the foundry may insert malicious circuits (hardware trojans) into the design without the designer's knowledge [8], [17]. Because of these attacks, the semiconductor industry loses $4 billion annually [19].

A. Split manufacturing

Leading fabless semiconductor companies such as AMD and research agencies such as Intelligence Advanced Research Projects Agency (IARPA) have proposed **split manufacturing** to thwart such attacks [10], [11]. In split manufacturing, the layout of the design is split into the Front End Of Line (FEOL) layers and Back End Of Line (BEOL) layers which are then fabricated separately in different foundries. The FEOL layers consist of transistors and other lower metal layers (≤M4) and the BEOL layers consist of the top metal layers (¿M4). Post fabrication, the FEOL and BEOL wafers are aligned and integrated together using either electrical, mechanical, or optical alignment techniques. The final ICs are tested upon integration of the FEOL and BEOL wafers [10], [11]. The asymmetrical nature of the metal layers facilitates split manufacturing. Figure 1 shows the cross-section of an IC and the pitches of different metal layers for the 45nm technology [1]. The top BEOL metal layers are thicker and have a larger pitch than the bottom FEOL metal layers. Hence, a designer can easily integrate the BEOL and FEOL wafers.



(a)

Layer	Pitch (nm)
M10	1600
M9	1600
M8	800
M7	800
M6	280
M5	280
M4	280
M3	140
M2	140
M1	130
Poly	125

(b)

Fig. 1: (a) A cross-section of an IC layout. The layout has two parts: Front End Of Line or FEOL layers(transistors, lower metal layers) and Back End Of Line or BEOL layers (intermediate, and top metal layers)[Source: [10]]. (b) Pitch length of different metal layers in 45nm CMOS technology [1].

Figure 2 shows a possible split manufacturing aware IC design flow. A gate level netlist is partitioned into blocks which are then floorplanned and placed. The transistors and wires inside a block form the FEOL layers. The top metal wires connecting the blocks and the IO ports form the BEOL layers. The BEOL and FEOL wires are assigned to different metal layers and routed such that the wiring delay and routing congestion are minimized. The layout of the entire design is split into two — one layout just contains the FEOL layers and the other layout just contains the BEOL layers. The two layouts are then fabricated in two different foundries.

In one embodiment, the fabricated FEOL and BEOL layouts are obtained by the System Integrator, and are then integrated by using electrical, mechanical, or optical alignment techniques and tested for defects. [11]. In another embodiment, the FEOL layout is first fabricated and then sent to a trusted second foundry where the BEOL layout is built on top of it [11].

Split manufacturing aims to improve the security of an IC as the FEOL and BEOL layers are fabricated separately and combined post fabrication. This prevents a single foundry (especially the FEOL foundry) from gaining full control of the IC. For instance, without the BEOL layers, an attacker in the FEOL foundry can neither identify the 'safe'places within a circuit to insert trojans nor pirate the designs without the BEOL layers. The economic benefit of split manufacturing comes from performing the low cost BEOL layer fabrication in-house and outsourcing the expensive FEOL layer fabrication [10]

B. Challenges in split manufacturing

Transporting the FEOL wafers to the BEOL foundry or transporting the FEOL and BEOL wafers to the SoC integrator

[1]companies that do not own their foundry are termed 'fabless' design houses.

978-3-9815370-0-0/DATE13/©2013 EDAA

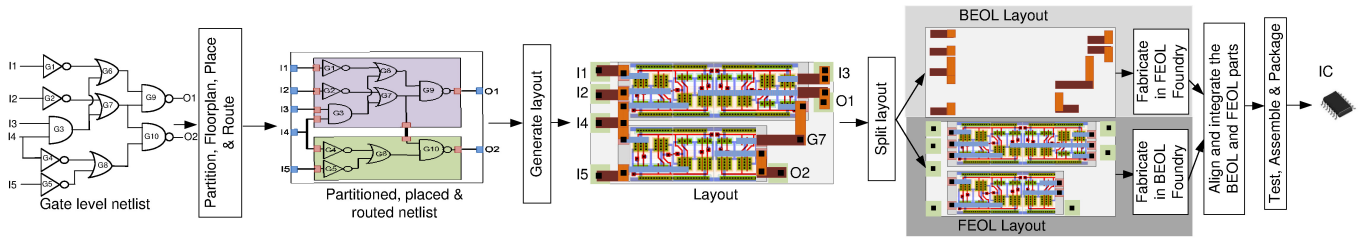


Fig. 2: Split manufacturing aware design flow.

may present a challenge; these wafers are thin and might crack or delaminate during transportation. Alignment of the FEOL and BEOL layers and increase in die area to accommodate alignment structures present another challenge. Split manufacturing may also affect the signal integrity, timing of the signals that span the FEOL and BEOL layers, and other design-for-manufacturability aspects. While several research projects from research agencies such as IARPA [10] and companies such as AMD [11] focus on addressing these challenges and make it feasible to reap the benefits of split manufacturing, we show that split manufacturing is inherently insecure.

C. Contributions

In this work, we address the following questions:

1) Is split manufacturing inherently secure?
2) If not, how can it be compromised?
3) How can it be strengthened?

The security offered by split manufacturing stems from the fact that the attacker in the FEOL foundry cannot determine the missing BEOL connections. We perform a security analysis of split manufacturing and show how an attacker can determine the missing BEOL connections by using knowledge of the FEOL connections. We propose *proximity attack* that exploits the vulnerabilities introduced by the physical design tools (floorplanning, placement, and routing tools). Then, the attacker can determine the missing BEOL connections and he can either pirate the design or insert trojans into it. We propose a defense to thwart the proximity attack by deceiving an attacker to make wrong BEOL connections. This technique involves the adoption of IC testing principles (fault excitation, fault propagation, and fault masking) to swap partition pins and improve security of split manufacturing.

II. Security Analysis of Split Manufacturing

A. Threat model

The attacker is in the offshore foundry that manufactures the FEOL part. Since the attacker has the GDSII layout file of the design, he/she can reverse engineer it and obtain the gate-level netlist. Such reverse engineering techniques have been demonstrated in [21]. The attacker in the FEOL foundry gains knowledge about most of the design (the transistors and the lower metal layers) except for the missing BEOL connections. Once the attacker determines these missing BEOL connections, he/she can reconstruct the original design.

B. Motivational example

Consider the ISCAS-85 combinational logic benchmark circuit, C17 shown in Figure 3. This design has five inputs (I1-I5) and two outputs (O1, O2). It is partitioned into two partitions – A (light colored) and B (dark colored). Partition A has four inputs (inputs of G1, G2, and G3) and two outputs (outputs of G7 and G9). Partition B has three inputs (inputs of G4, G5, and one input of G10) and one output (output of G10). Traditionally, the wires within a partition (local wires except Vdd and clock) are assigned to lower metal layers. The wires that span the partitions and I/O ports are assigned to higher metal layers. This makes the routing easier [20]. The nets connecting the input ports I1-I5 to the corresponding inputs of the gates G1-G5 use the BEOL layers. The nets connecting the output of gates G9 and G10 to output ports O1 and O2, respectively, use the BEOL layers. The net that connects the output of G7 to one of the inputs of G10 also uses the BEOL layer.

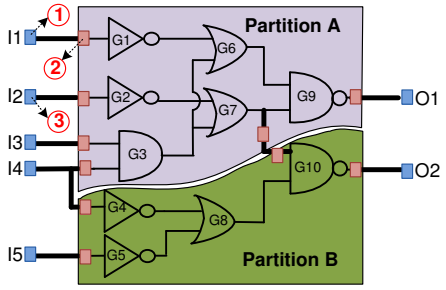


Fig. 3: The C17 benchmark circuit. The violet (light) and green (dark) regions represent partitions A and B, respectively. The output of G7 goes between the partitions. Thin lines denote FEOL nets while the thick lines denote the BEOL nets. ① Target pin $P_{I1,IO,in}$, ② corresponding candidate pin $P_{I1,A,in}$, and ③ swapping pin $P_{I2,IO,in}$.

C. Terminology

In this paper, $P_{Net,Partition,Direction}$ denotes a partition pin or an IO port. Net is the name of the wire in the original design. $Partition$ represents either the partitions A or B or the IO port. $Direction$ of a pin can be either in or out. Consider a net X in the original design which connects a gate in Partition A to another gate in Partition B. The corresponding partition pins on the partition boundaries of Partition A and Partition B are denoted as $P_{X,A,out}$ and $P_{X,B,in}$, respectively.

A *target pin* is an output pin of a partition or an input port of the design from which a signal originates. A *candidate pin* is an input pin of a partition or an output port of the design at which a signal terminates. Consider the C17 circuit shown in Figure 3. Pin $P_{I1,IO,in}$ is the target pin and pin $P_{I1,A,in}$ is its corresponding candidate pin. A target pin in a partition will be swapped with another pin in the same partition referred to as *swapping pin*. In Figure 3, pin $P_{I2,IO,in}$ is the swapping pin for target pin $P_{I1,IO,in}$.

Every missing BEOL connection is a net that connects a target pin and its corresponding candidate pin. A target pin has many candidate pins but the attacker is trying to determine the correct candidate pin for that target pin with the following objective: If she can connect every target pin with its correct candidate pin, she can recover the original design.

D. Proximity attack

The proposed attack is based on the heuristic that floorplanning and placement (F&P) tools place the partitions close by and orient the partitions so as to reduce the wiring (delay) between the pins to be connected [20]. *This heuristic of most F&P tools constitute a security vulnerability that can be exploited by an attacker in the FEOL foundry who does not have access to the BEOL layers.*

Consider a target pin, $P_{X,A,out}$, and its corresponding candidate pin, $P_{X,B,in}$. F&P tools will try to place $P_{X,A,out}$ closer to $P_{X,B,in}$ than to any other partition pin in Partition B. An attacker may then recover the netlist of the original design by connecting every target pin to its closest candidate pin. This is referred to as the *proximity attack*. This attack uses the hints provided by the F&P tools that are explained below:

Hint 1 – Input-output relationships: An input partition pin (candidate pin) will be connected either to an output pin of another partition or to an input port of the IC (target pin). Input partition pins are usually connected to the poly layer and output partition pins usually emanate from the diffusion layer.

Consider the partition pin $P_{I1,A,in}$ of the partitioned F&P C17 benchmark circuit in Figure 3. This is an input pin for Partition A. This pin can be connected only to pins $P_{I1,IO,in}$, $P_{I2,IO,in}$, $P_{I3,IO,in}$, $P_{I4,IO,in}$, $P_{I5,IO,in}$, and $P_{G10,B,out}$, reducing the possible candidate pins from 10 to 6.

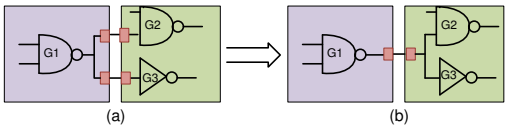


Fig. 4: Typical fan-out push performed by F&P tools.

Hint 2 – Unique inputs per partition: A net in a design will be connected to only one input pin of a partition. If a net acts as an input for multiple gates within that partition, the fan-out node is usually placed within the partition that it feeds into.

Consider the partition in Figure 4. Partition A has two output pins $P_{G1,A,out1}$ and $P_{G1,A,out2}$ and Partition B has two input pins $P_{G1,B,in1}$ and $P_{G1,B,in2}$. On pushing the fan-out node into Partition B, as shown in Figure 4(b), the number of input and output pins in Partition A and Partition B, respectively, is reduced by one. This observation reduces the number of possible connections from 2^{N^2} to N! for N missing connections, as every target pin will have a unique candidate pin.

Hint 3 – Combinational loops: With the exception of ring oscillators, flip-flops, and latches, combinational loops are rare in a design. Furthermore, ring oscillators, flip-flops, and latches are contained within a single partition and are easily identifiable due to their standard structure. Therefore, an attacker does not need to consider a pin as a candidate pin, if it forms a combinational loop with the target pin.

Consider the partitions in Figure 3. After connecting the target pin $P_{G7,A,out}$ to candidate $P_{G7,B,in}$, an attacker will not consider the output pin $P_{G10,B,out}$ as a possible candidate pin for the target pins $P_{I2,A,in}$, $P_{I3,A,in}$, and $P_{I4,A,in}$ because $P_{G10,B,out}$ will form a combinational loop when connected to any one of these pins.

An attacker can find the correct candidate pin for a target pin by identifying the closest pin from the list of possible candidate pins. As mentioned before, this heuristic is based on the fact that F&P tools try to place two partition pins, which are to be connected by a BEOL layer, as close as possible to each other to reduce the wiring overhead. So, an attacker can connect the two closest pins in different partitions hoping that F&P tools must have placed them close to each other.

Consider the locations of partition pins and IO ports of the F&P C17 benchmark as shown in Table I. Consider the input port $P_{I1,IO,in}$ which is connected to pin $P_{I1,A,in}$ in partition A. The locations of $P_{I1,IO,in}$ and $P_{I1,A,in}$ are (0,6) and (1,6), respectively. The distance between these two pins is 1 unit. Now, consider another input port $P_{I3,IO,in}$. The distance between $P_{I3,IO,in}$ and $P_{I1,A,in}$ is 1.414 units. Thus, the closest possible pin to $P_{I1,A,in}$ is $P_{I1,IO,in}$. Hence, an attacker will connect these two pins in the netlist and obtain the missing BEOL connection. Similarly, he can connect all the other partition pins with their closest pins and reconstruct the original design.

TABLE I: X-Y coordinates of the pins in partitions A and B and IO ports of F&P C17 design. The coordinates are shown as absolute units for ease of understanding.

Partition A		Partition B		Input & Output	
Pin	XY location	Pin	XY location	Port	XY location
$P_{I1,A,in}$	(1,6)	$P_{I4,B,in}$	(1,2)	$P_{I1,IO,in}$	(0,6)
$P_{I2,A,in}$	(9,5)	$P_{I5,B,in}$	(1,0)	$P_{I2,IO,in}$	(10,6)
$P_{I3,A,in}$	(1,5)	$P_{G7,B,in}$	(7,2)	$P_{I3,IO,in}$	(0,5)
$P_{I4,A,in}$	(1,4)	$P_{G10,B,out}$	(7,0)	$P_{I4,IO,in}$	(0,4)
$P_{G9,A,out}$	(9,4)			$P_{I5,IO,in}$	(0,0)
				$P_{O1,IO,out}$	(10,5)
				$P_{O2,IO,out}$	(8,0)

E. Proximity attack algorithm

Input : FEOL layers
Output: Netlist with BEOL connections
Reverse engineer FEOL layers and obtain the partitions;
while *Unassigned partition pins or ports exist* **do**
  Select an arbitrary unassigned input pin or an output port as a $TargetPin$;
  $ListOfCandidatePins$ = BuildCandidatePinsList($TargetPin$);
  Select $CandidatePin$ from $ListOfCandidatePins$ that is closest to $TargetPin$;
  Connect $TargetPin$ and $CandidatePin$;
  Update netlist;
end
Return netlist;

BuildCandidatePinsList(TargetPin)
Input : $TargetPin$ $P_{X,i,in}$
Output: Candidate pins for $TargetPin$
CandidatePins = Unassigned output pins of other partitions + unassigned input ports of the design;
For each $PinJ \in$ *CandidatePins* **do**
  if *CombinationalLoop(TargetPin, PinJ)* **then**
    CandidatePins -= PinJ;
  end
end
Return CandidatePins;

Algorithm 1: Proximity attack on split manufacturing

Algorithm 1 describes the steps involved in the proximity attack. The input to the algorithm is the FEOL layer information and the goal is to reconstruct the netlist by identifying the missing BEOL connections[2]. The algorithm chooses an arbitrary $TargetPin$ from the unassigned partition input pins and output ports, creates its list of possible $CandidatePins$, and connects it to the closest pin in this list. The netlist is then updated. This procedure is repeated until all the missing connections are made. Candidate pins for a target pin are chosen based on the observations 1–3. After executing this algorithm, the attacker obtains the missing BEOL connections and, consequently, the original design.

III. Secure Split Manufacturing

One technique to overcome proximity attack is to rearrange the partition pins such that a pin $P_{X,A,out}$ will no longer be the closest pin to $P_{X,B,in}$. An attacker performing proximity attack will be deceived into making the wrong BEOL connections, i.e., $P_{X,B,in}$ will be connected with $P_{Y,A,out}$ instead of $P_{X,A,out}$.

Consider the F&P C17 circuit. Before swapping, pin $P_{G7,A,out}$ is close to pin $P_{G7,B,in}$. Hence, an attacker will connect them. If the pins $P_{G7,A,out}$ and $P_{G9,A,out}$ are swapped, then $P_{G9,A,out}$ will be close to $P_{G7,B,in}$. Therefore, an attacker will connect these two pins, thereby making a wrong connection.

Objective: Sufficient number of pins have to be swapped such that the functionality of the deceiving netlist[3] differs from that of the original netlist. This functional difference can be quantified by the Hamming distance between the outputs of the original netlist and the deceiving netlist. If it is 0%, then the attacker is able to retrieve the original design. If it is 100%, then the attacker is able to retrieve the design that is the exact complement of original design. Hence, ideally, the Hamming distance should be 50% where a different set of the outputs are corrupted for different input vectors. A designer can stop swapping pins when the Hamming

[2]The netlist of the FEOL layout [21] can be obtained from the GDSII netlist of the FEOL part. Several tools for this purpose exist [7].

[3]Deceiving netlist is the netlist that the defender manipulates by swapping the pins.

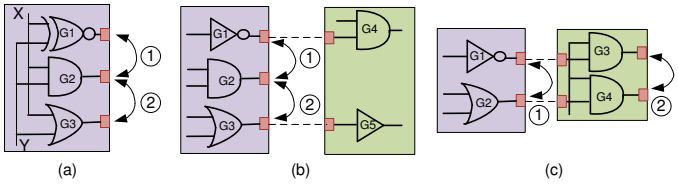


Fig. 5: IC testing principles applied to split manufacturing: (a) Pin that has a logical value opposite to that of the swapped pin is preferred. ① Values at $P_{G1,A,out}$ and $P_{G2,A,out}$ differ only when X=Y=0; ② Values at $P_{G2,A,out}$ and $P_{G3,A,out}$ differ in two cases: X=1, Y=0 and X=0, Y=1. Thus, $P_{G3,A,out}$ is selected as the swapping pin for $P_{G2,A,out}$. (b) If $P_{G1,A,out}$ is selected as the swapping pin for $P_{G2,A,out}$, the wrong value will propagate only when the other input of G4 is 1. However, if $P_{G3,A,out}$ is selected as the swapping pin, the buffer, G5, will always propagate the wrong value. (c) The logical error introduced by swapping $P_{G1,A,out}$ and $P_{G2,A,out}$ is canceled by swapping $P_{G3,B,out}$ and $P_{G4,B,out}$.

distance between the outputs of the original netlist and the deceiving netlist reaches 50%. Finding the best rearrangement for N pins of a partition might take N! computations and this is computationally expensive. We will consider pair-wise swapping of pins. Pair-wise swapping of pins results in $O(N^2)$ computations.

Constraints on pin swapping: Not all pins can be swapped with all other pins. The target pin and swapping pin together should pass the basic tests presented as hints in the previous section. Otherwise, the attacker might omit it from further consideration. Therefore, for a target pin, a swapping pin should

- be an output pin of the partition where the target pin resides,
- not be connected to the partition where the candidate pin resides, and
- not form a combinational loop with a candidate pin on connecting with it.

A. IC testing principles for split manufacturing

To find a swapping pin for a target pin, similar to an attacker, the defender can build the list of candidate pins for that target pin. Then, he/she can randomly select the swapping pin from that list. Unfortunately, such random selections might not guarantee that the attacker will get a wrong output on making a wrong connection. Hence, we use IC testing principles [5], to select the swapping pin for a target pin in order to achieve the 50% Hamming distance objective.

Scenario 1 – Commutativity: Consider the scenario where the swapping pin and the target pin are the two inputs of the same gate that implements a commutative operation, and neither of them act as an input of any other gate. On swapping these two pins, the logical functionality remains the same despite the wrong connection.

Scenario 2 – Fault activation: The logical values at the swapping pin and target pin should differ for most of the input patterns. If their logical values are the same for most of the input patterns, then the resulting design, even with wrong connections, will still produce mostly correct outputs. This is similar to fault activation in IC testing where, in order to detect a stuck-at-fault at a node, the node should be justified to the value that is the opposite of the stuck-at value [5]. Thus, a pin that has a logical value opposite to that of the target pin for most of the input patterns should be selected as the swapping pin.

Consider the partitioned design shown in Figure 5 (a). If $P_{G1,A,out}$ is selected as the swapping pin for target pin $P_{G2,A,out}$, the logical values at the pins differ when X=Y=0. If $P_{G3,A,out}$ is selected as the swapping pin for target pin $P_{G2,A,out}$, the logical values at the pins differ in two cases: X=1, Y=0 and X=0, Y=1. Thus, $P_{G3,A,out}$ is preferred over $P_{G1,A,out}$.

Scenario 3: Fault propagation: Pins should be swapped such that a wrong value activated by the swap can easily propagate to one or more outputs and corrupt them. If the swapping pin results in a wrong value which does not propagate to one or more outputs, then that swap is ineffective. Once again, this is similar to the fault propagation concept in IC testing where the effect of a fault has to propagate to one or more outputs for detection [5]. Thus, pins should be swapped such that the effect of swapping propagates to one or more outputs.

Consider the partitioned design shown in Figure 5 (b). If $P_{G1,A,out}$ is selected as the swapping pin for $P_{G2,A,out}$, the wrong value will propagate only when the other input of G4 is 1. However, if $P_{G3,A,out}$ is selected as the swapping pin, the buffer, G5, will always propagate the wrong value. Thus, $P_{G3,A,out}$ is preferred over $P_{G1,A,out}$.

Scenario 4: Fault masking: Sometimes, logical values corrupted by swapping pins in partition A can be restored to their original value because of swapping pins in partition B. This is similar to fault masking in IC testing where the effect of one fault is restored by the effect of another fault [5].

Consider the partitioned design shown in Figure 5 (c). The logical error introduced by swapping $P_{G1,A,out}$ and $P_{G2,A,out}$ is canceled by swapping $P_{G3,B,out}$ and $P_{G4,B,out}$.

Thus, random selection of a swapping pin might not guarantee a wrong output for an attacker while increasing the wire length for the defender. Hence, we propose a judicious swapping technique based on fault analysis, which is proposed in [16].

Input : Partitions
Output: List of target and swapping pins
$ListofTargetPins = \phi$;
$ListofSwappingPins = \phi$;
$ListofUntouchedPins$ = *All partition pins and I/O ports*;
while *Untouched output partition pins or input ports exist* **do**
  For each $UntouchedPin$ **do**
    $SwappingPins$ = BuildSwappingPinsList($UntouchedPin$);
    For each $SwappingPin \in SwappingPins$ **do**
      Compute $FaultImpact_{UntouchedPin,SwappingPin}$;
    end
  end
  Find the $TargetPin$ and $SwappingPin$ with the highest $FaultImpact$ from its $SwappingPins$;
  $ListofTargetPins$ += $TargetPin$;
  $ListofSwappingPins$ += $SwappingPins$;
  $ListofUntouchedPins$ -= $TargetPin$;
  $ListofUntouchedPins$ -= $SwappingPin$;
  Swap $TargetPin$ and $SwappingPin$;
  Update netlist;
end
Return $ListofTargetPins$ and $ListofSwappingPins$;

BuildSwappingPinsList($TargetPin$);
Input : $TargetPin$ $P_{X,i,out}$
Output: $SwappingPins$ for $TargetPin$
$SwappingPins$ = Other untouched output pins in partition 'i';
For each $PinJ \in SwappingPins$ **do**
  if *CombinationalLoop($TargetPin$, PinJ)* **then**
    $SwappingPins$ -= PinJ;
  end
end
Return $SwappingPins$;

Algorithm 2: Fault analysis-based swapping of pins to thwart proximity attack

B. Defense: Fault-analysis based pin swapping

Instead of randomly selecting the swapping pin, the pin that affects most of the outputs for most of the input patterns on swapping is selected. This accounts for fault activation, propagation, and masking scenarios. We define the fault impact

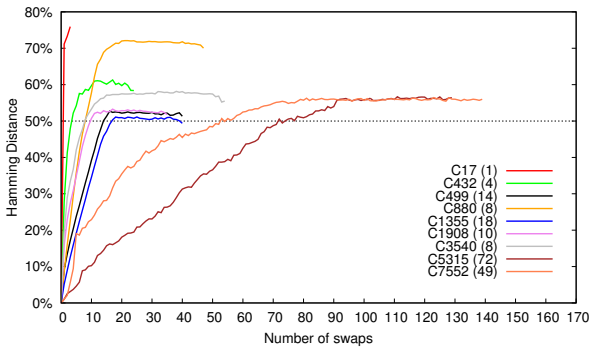


Fig. 6: Hamming distance between the outputs of original design and the design obtained by swapping partition pins based on fault analysis technique. The number of swaps required to achieve the 50% Hamming distance is shown in parenthesis next to the circuit name.

metric, to select a swapping pin Y for a target pin X,

$$\text{Fault impact}_{X,Y} = \sum_{i=1}^{\text{\# of test patterns}} \text{\# of corrupted outputs}$$

We swap the target pin X with the swapping pin Y in the netlist and identify the cumulative sum of the corrupted output bits over a set of random test patterns. Fault impact quantifies the effect of swapping on the outputs of the design.

Pins can be swapped based on the fault impact metric as shown in Algorithm 2. For an untouched pin (all output partition pins and input ports), a list of swapping pins, $SwappingPins$, is built using the observations 1–3 in Section II. Fault impact metric is used to select the swapping and target pins. The selected pins are then swapped and the netlist is updated. The above steps are repeated until all the partition pins and input ports are swapped or the Hamming distance value reaches 50%.

IV. Results

A. Experimental setup

The proposed technique was evaluated using ISCAS-85 combinational benchmark circuits[4]. Each circuit was partitioned into two partitions using the hMETIS tool [18]. Floorplanning, placement, and routing were performed using Cadence SoC Encounter tool [6] for 45nm CMOS technology. The location of the partition pins and IO ports were obtained using the same tool. We used the HOPE fault simulation tool [13] to calculate the fault impact metric by applying 1000 random input patterns. The Hamming distance between the output of the original design and the design reconstructed using the proximity attack technique was determined by applying 1000 random input patterns. The defender stops swapping pins once he/she reaches the 50% Hamming distance between the original netlist and the deceiving netlist constructed by swapping pins. In case of designs where 50% Hamming distance was not achieved, he/she swaps all the pins.

B. Required number of swaps

The purpose of swapping pins is to ensure that an attacker on performing a proximity attack reconstructs an incorrect design, i.e., the reconstructed design produces wrong outputs for most of the inputs. The Hamming distance metric not only quantifies the tendency of a design to produce a wrong output but also quantifies how many output bits are corrupted. Figure 6 shows the Hamming distance between the outputs of the original design and the deceiving netlist obtained after swapping partition pins based on the proposed fault analysis.

[4]The physical design benchmarks were not used as they lack information about the functionality of the gates, which is important to analyze the effectiveness of the proposed techniques in producing a wrong output.

Different benchmark circuits have different numbers of partition pins. Thus, only a limited number of swaps is possible in a circuit. In addition, based on the order of swapping, some partition pins might not have candidate pins due to the observations 1–3 in Section II. For instance, some of the swapping pins might form a combinational loop with the candidate pin, leaving no swapping possibilities.

Fault analysis-based swapping achieves at least 50% Hamming distance for all the benchmarks. This is because, it accounts for the fault activation, propagation, and masking effects in pin-swap selections. Furthermore, the curves are steep in fault analysis-based swapping. This indicates that fault analysis-based swapping will take a small number of swaps to achieve the 50% Hamming distance mark.

C. Attack metric: Number of correct connections

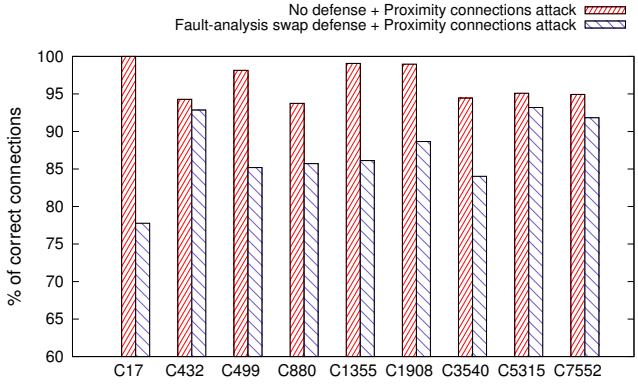


Fig. 7: Percentage of partition pins and IO ports that are correctly connected by an attacker using proximity attack.

Naturally, an attacker tries to make as many correct connections as possible. The number of correct connections in a design reconstructed by the attacker will determine the effectiveness of the proximity attack. Figure 7 shows the percentage of partition pins and IO ports that are correctly connected by an attacker using proximity attack.

In case of 'No defense + Proximity connections attack', the average number of correct connections is 96%. In case of the C17 benchmark, the attacker is able to make all the connections correctly. This verifies our earlier claim that F&P tools place pins, which will be connected in BEOL, closer to each other and indicates that straightforward split manufacturing can be easily compromised. In case of 'Fault-analysis swap defense + Proximity connections attack', the attacker connects 87% of the pins correctly because only a small number of pins were swapped; for most of the designs at most 20 pins were swapped. However, the effectiveness of the remaining 13% of wrong connections will be explained below.

D. Defense metric: Hamming distance of designs reconstructed by the attacker

Figure 8 depicts the Hamming distance between the outputs of the original design and the design reconstructed using proximity attack. When proximity attack is performed, the Hamming distance between the outputs of the original and reconstructed designs reduces which highlights the effectiveness of the proposed attack. This is particularly evident in the case where no defense is used.

In case of 'No defense + Proximity connections attack', the average Hamming distance is around 10% except for C7552 circuit. In this circuit, there are more number of I/O ports shared between the two partitions and are placed at equidistant locations. This prevented the attacker from making the correct connections, resulting in a higher Hamming distance. However, for most of the benchmark circuits, the Hamming distance value of the design reconstructed by the attacker is less than 6%. Thus, an attacker almost determines the functionality of the design correctly using this attack. Consequently, he/she

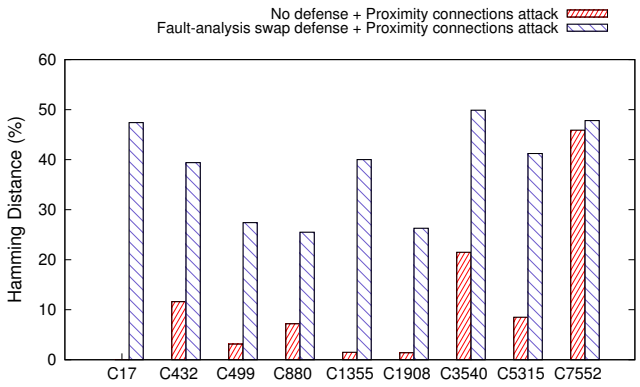


Fig. 8: Hamming distance between the outputs of the original design and the design reconstructed using proximity attack.

obtains a design which is almost an equivalent to the original design. This necessitates the need for a defense.

Fault analysis-based swapping achieves a Hamming distance that is close to 50% Hamming distance objective. This illustrates that by swapping a limited but an effective set of pins, a designer can obtain a secure design. The average Hamming distance for 'Fault-analysis swap defense + Proximity connections attack' technique is 42%. This is slightly less than the desired 50% Hamming distance distance because the fault-analysis based defense does not consider the proximity of the swapping and target pins. In other words, the proposed defense places the swapping pin closer to the candidate pin than the target pin. However, there might be another pin, apart from the swapping and target pins, that is closer to the candidate pin. Thus, an attacker will connect the candidate pin with this pin instead of the swapping pin that the defender had in mind. Nevertheless, employing the fault-analysis based defense improves the Hamming distance significantly.

Using the percentage of correct connections as a metric for a defense is a fallacy. This is evident from the C3540 circuit. Consider the 'Fault-analysis swap defense + Proximity connections attack' in Figure 7. Here, the attacker makes 85% of connections correctly using proximity attack. Consider the same case in Figure 8. Here, the fault analysis based defense achieves 50% Hamming distance. Thus, a small number of wrong connections are enough to corrupt majority of the output bits because of the reasons stated in Section III.

Attack metric versus defense metric: To summarize, the number of correct connections made is a good metric to analyze an attack technique because the objective of an attacker is to get more number of connections correctly. On the contrary, the Hamming distance between the outputs of the original design and the design constructed by performing an attack is a good metric to analyze a defense because the objective of the defender is to deceive an attacker into making wrong connections such that a large number of wrong outputs are obtained.

V. Related work

Logic obfuscation is another technique that protects IPs by hiding the functionality and the implementation of a design by adding (modifying) the components into (in) the original design [8], [17]. Gates and flip-flops are added into the design for this purpose. In order for the design to function correctly (i.e., produce correct outputs), a valid key has to be supplied to these gates/ flip-flops [8], [17]. If a wrong key is applied, the obfuscated design will exhibit a wrong functionality (i.e., produce wrong outputs). A circuit can also be obfuscated by replacing gates with memory elements [4].

Techniques that target detection of trojans inserted at a foundry include power measurements [3], delay measurements [12], gate-level characterization [15], and a combination of these techniques [14]. In contrast, split manufacturing is a proactive technique that prevents insertion of trojans.

VI. Conclusion

Split manufacturing is not a universal solution for all security problems. It can protect commercial designs from rogue elements in the FEOL foundry. In this work, we have shown how an attacker can use the objective of F&P tools to undermine the security benefits offered by split manufacturing. When no defense is applied in conjunction with split manufacturing, the attacker is able to make 96% of the missing BEOL connections correctly.

While the proposed pin swapping technique can increase the wire-length, noise, and reduce signal integrity, we showed that only a small set of pins (¡20 for most designs) have to be swapped to achieve the 50% Hamming distance metric. For high-performance designs, one can easily constrain the proposed algorithm to not to consider pins on the critical path. Even though the proposed techniques have been demonstrated using only two partitions, one can easily extend them to multiple partitions. Apart from swapping pins, one can also partition the design, determine the aspect-ratios, and orient the partition blocks with an objective to overcome proximity attack. The floorplanning, placement, and routing tools can also be empowered with security heuristics.

References

[1] FreePDK45:Metal Layers. http://www.eda.ncsu.edu/wiki/FreePDK45:Metal_Layers.

[2] International Technology Roadmap for Semiconductors. http://www.itrs.net/Links/2011ITRS/Home2011.htm.

[3] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar. Trojan Detection using IC Fingerprinting. *Proc. of the IEEE Symposium on Security and Privacy*, pages 296–310, May 2007.

[4] A. Baumgarten, A. Tyagi, and J. Zambreno. Preventing IC Piracy Using Reconfigurable Logic Barriers. *IEEE Design and Test of Computers*, 27(1):66–75, 2010.

[5] M. L. Bushnell and V. D. Agrawal. Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI Circuits. *Kluwer Academic Publishers, Boston*, 2000.

[6] Cadence. SoC Encounter. http://www.cadence.com/products/di/soc_encounter/pages/default.aspx.

[7] Cadence. Virtuoso. http://www.cadence.com/products/cic/layout_suite/pages/default.aspx.

[8] R.S. Chakraborty and S. Bhunia. HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection. *IEEE Transactions on Computer-Aided Design*, 28(10):1493–1502, 2009.

[9] DIGITIMES Research. Trends in the global IC design service market. http://www.digitimes.com/Reports/Report.asp?datepublish=2012/3/13&pages=RS&seq=400&read=toc.

[10] Intelligence Advanced Research Projects Activity. Trusted Integrated Circuits Program. https://www.fbo.gov/utils/view?id=b8be3d2c5d5babbdffc6975c370247a6.

[11] R.W Jarvis and M. G. McIntyre. Split manufacturing method for advanced semiconductor circuits. *US Patent no. 7195931*, 2004.

[12] Yier Jin and Y. Makris. Hardware Trojan detection using path delay fingerprint. *Proc. of IEEE International Symposium on Hardware-Oriented Security and Trust*, pages 51–57, 2008.

[13] H.K. Lee and D. S. Ha. HOPE: An Efficient Parallel Fault Simulator for Synchronous Sequential Circuits. *IEEE Transactions on Computer-Aided Design*, 15(9):1048–1058, 1996.

[14] S. Narasimhan, Du Dongdong, R.S. Chakraborty, S. Paul, F. Wolff, C. Papachristou, K. Roy, and S. Bhunia. Multiple-parameter side-channel analysis: A non-invasive hardware Trojan detection approach. *Proc. of the IEEE International Symposium on Hardware-Oriented Security and Trust*, pages 13–18, 2010.

[15] Miodrag Potkonjak, Ani Nahapetian, Michael Nelson, and Tammara Massey. Hardware Trojan horse detection using gate-level characterization. *Proceedings of the IEEE/ACM Design Automation Conference*, pages 688–693, 2009.

[16] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri. Logic Encryption: A Fault Analysis Perspective. *in the Proc. of Design Automation and Test in Europe*, pages 953–958, 2012.

[17] J.A. Roy, F. Koushanfar, and I.L. Markov. EPIC: Ending Piracy of Integrated Circuits. *Proc. of Design, Automation and Test in Europe*, pages 1069–1074, 2008.

[18] N. Selvakkumaran and G. Karypis. Multiobjective hypergraph-partitioning algorithms for cut and maximum subdomain-degree minimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2006.

[19] SEMI. Innovation is at risk as semiconductor equipment and materials industry loses up to $4 billion annually due to IP infringement. www.semi.org/en/Press/P043775, 2008.

[20] Naveed A. Sherwani. Algorithms for VLSI Physical Design Automation. 2002.

[21] R. Torrance and D. James. The state-of-the-art in semiconductor reverse engineering. *Proc. of IEEE/ACM Design Automation Conference*, pages 333–338, 2011.