Fall 2018

Hardware Trojan Attack and Defense Techniques

Aman Gupta

Hardware Trojan Attack and Defense Techniques

By

Aman Gupta

A creative component report submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Electrical Engineering

Program of Study Committee:

Dr Randall L. Geiger, Major Professor

Iowa State University

Ames, Iowa

2018

Contents

List of Figures

List of Tables

Acknowledgement

I would like to express my gratitude to my Major professor, Dr Randall L. Geiger for showing me the right path and always supporting me throughout my journey.

In addition, I would also like to thank my friends, colleagues, the department faculty and staff for making my time at Iowa State University a wonderful experience.

Abstract

Hardware Trojans have become a significant threat to ICs/SoCs used in defense or other security critical applications. This threat is becoming more prominent because majority of the chip design companies rely upon third parties for fabricating their design. This is due to the higher fabrication costs involved as the technology node is shrinking. This leads to Fabrication-time attacks since an adversary at the untrusted fabrication house can make malicious modifications to the design. This may lead to reduced lifecycle of a chip, leaking out secret information et cetera. While there is a diverse set of Hardware Trojans available, we need the Trojan to be as stealthy as possible, having a smaller area, causing minimal perturbations to the existing design in terms of power, temperature & delay, and having a complex triggering mechanism. In this paper, we will study one such Hardware Trojan attack. Detecting Hardware Trojans is very challenging due to the diversity of Trojans. We would also look at different ways in which a Trojan attack can be detected. Then, we would evaluate if any of those defenses can detect our example attack.

Chapter 1. Introduction

The technology node for IC's is shrinking to deep submicron levels and while this has been a boon for the performance, the complexity and cost of fabricating these ICs is increasing dramatically. According to data available online[1], it costs about 15% more to set up the fabrication lab for each successive process node. Due to the high costs involved, most of the chip design companies do not opt for an in-house fabrication. Instead, a lot of design houses are becoming fabless i.e. they outsource the fabrication of their IC's and rely upon a third party to fabricate their design. This gives opportunities to the attackers to make malicious modifications to the original design, known as Hardware Trojans. These Trojans pose a serious threat to security critical applications by causing chip malfunctions, reducing the reliability/lifetime of the ICs, leak out secret information to the attackers. HTs can be inserted into a design at any phase from RTL Design to Fabrication. Figure 1[2] below depicts different phases in the IC design cycle where a Trojan can be inserted by rogue engineers. Depending upon the phase where Trojan is inserted, the attack can be categorized as a Design Time Attack or a Fabrication Time Attack. The likelihood of a Fabrication time attack being detected is relatively less since it has to go through a fewer verification stages, namely, Post-Fabrication Testing as compared to the Design Time attacks. Therefore, in this paper, we would be focusing upon Fabrication time attacks.

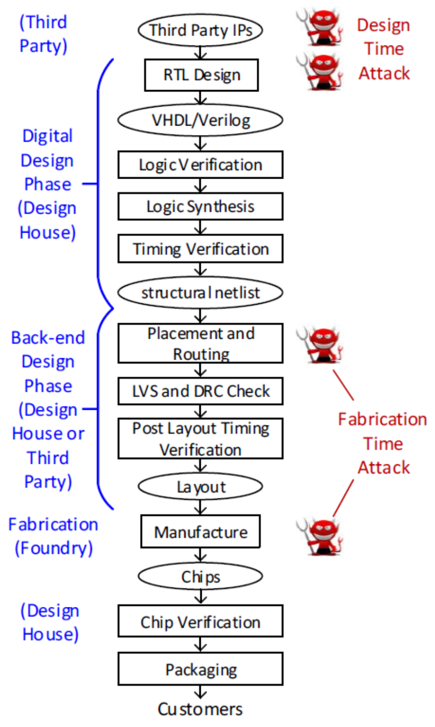


Figure 1. 1

<u>Different phases in IC Design Cycle where a HT can be inserted</u>

One of the well-known fabrication time attack is the Dopant-Level Trojan[3],[4]. These dopant-level trojans make the existing circuitry malicious by identifying the victim transistors and changing the dopant ratio on the input pins. They tie the input of the victim transistors to a logic 0 (Open Circuit) or Logic 1 (Short Circuit). These trojans are very hard to detect since they modify the existing circuitry and do not introduce any additional circuitry. To detect dopant-level trojans, a complete chip delayering and comprehensive imaging with a scanning electron microscope has to be done. But there is a trade-off between this elusiveness and the expressiveness of dopant-level trojans. Since, dopant level trojans do not have a sophisticated trigger, they are more detectable by post-fabrication testing. Therefore, we need a HT attack which is controllable, stealthy and small. In this paper, we would discuss about one such attack which requires as little as one gate in a sea of millions of gates, has a sophisticated trigger and has negligible perturbations on the original design in terms of area, power and delay. The attack works by sucking out charge from a victim wire every time it transitions and storing that charge on a capacitor. When the voltage across the capacitor becomes more than a threshold value, payload is deployed - which is changing the value of a victim flip-flop to any desired value.

To defend against Hardware Trojan attacks, researchers have proposed different methods like 1) using side channel information[5],[6](power, temperature, delay) to check for anomalous behavior, 2) adding on-chip sensors[7] to measure features like signal propagation delay, temperature changes and check if there are any dramatic changes in those features, 3) Split Manufacturing[8], 4) Addition of Built-In Self Authentication cells instead of non-functional filler cells[9]. These defense techniques will be explained further in the paper. We would then evaluate which of the defense techniques are capable of detecting our example attack and which ones are not. Lastly, we would review the strengths and shortcomings of our example attack.

Rest of the paper is organized as follows: Chapter 2 presents Hardware Trojan Overview and the Example Attack. In Chapter 3, we will discuss the Possible Defenses against the Hardware Trojans. In Chapter 4, we will check if our defenses are capable of detecting the HT in Example Attack. Chapter 5 will cover the strengths and shortcomings of our Example Attack. Finally, Chapter 6 will conclude the paper.

Chapter 2. Hardware Trojan Overview

2.1 What is a Hardware Trojan?

Hardware Trojan is any addition or modification to a circuit or a system with malicious intention. Hardware Trojans are persistent i.e. once a system has been infected, the threat remains every time the system is turned on. They have the potential ability to undermine confidence in all modern electronic systems.

2.2 Characteristics of a Hardware Trojan

1. Change or control the functionality of an Integrated Circuit(IC) or a System on Chip(SoC), for example changing the logic value of a security critical flipflop that has some serious implications.
2. A Hardware Trojan can assist in leaking sensitive information, for example by propagating internal signals to the output pins. These signals can disclose secret information to attackers.
3. Reduce Circuit Reliability, for example by introducing additional circuitry that is capable of creating local temperature hotspots in the IC and thereby, causing the chip to fail eventually.

2.3 Building Blocks of Hardware Trojans

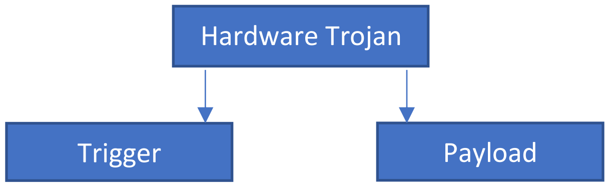


Figure 2. 1 *HT Building Blocks*

2.3.1 Trigger A Trigger is some circuitry that activates the attack.

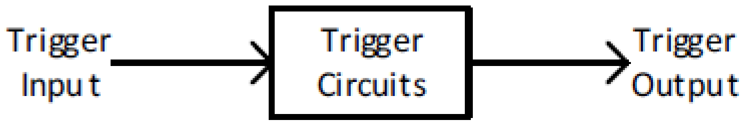


Figure 2. 2 *Trigger Circuit*

<u>**Characteristics of a good Trigger Circuit:**</u>

1. **Small Area:** The trigger circuit should be small enough so that it has better chances of escaping detection. For example, it should be small enough so that it can fit into the empty spaces left after the placement and routing of any design.
2. **Functionality:** The trigger circuit must be able to detect the toggling events of a target victim wire, serving as an input to the trigger circuit.

3. **Low Power:** Since trigger circuit is an additional circuitry, it should have low power consumption which can be well hidden within the normal fluctuations of entire chip's power consumption.
4. **Negligible Timing Perturbation:** The trigger circuit should not impact the existing path delays in the original design by a huge amount. The timing perturbations should not be easily separable from the noise common to the path delays.
5. **Standard Cell Compatibility:** The trigger circuit must be able to fit within the standard cell dimensions.

2.3.2 Payload A Payload is the actual action performed by the attack. For example, it might be changing the logical value of a security critical register or it can be dramatically increasing the temperature consumption of the entire chip.

2.4 Example Attack

In this attack[2], analog characteristics of integrated circuits are leveraged to implement the trigger. The attacker uses the Graphic Database System II (GDSII) file, which is a polygon representation of already placed and routed design. By doing some preprocessing steps, the attacker identifies the location of unused spaces in the laid-out design. These unused spaces are then used to fit in the Trojan circuitry, which includes the trigger and the payload. Assumption here is that these unused spaces are filled with dummy logic, also called as non-functional Filler cells. Removing these filler cells won't impact the functionality of the design.

This attack is a privilege escalation attack. Once the trigger circuit is activated, the payload is deployed. Here, payload is to change the privilege bit of a processor which escalates the privilege of a User Mode process to Supervisor Mode. This gives the attacker full control over the processor.

2.4.1 Trigger Circuit The trigger circuit[2] consists of a capacitor, which will siphon charge from the nearby victim wires as they transition between digital values. It essentially performs the analog integration of charge from a viable victim wire and is also able to reset itself through natural charge leakage. When enough charge is stored and the voltage across the capacitor is above a predefined threshold voltage, trigger circuit is activated.

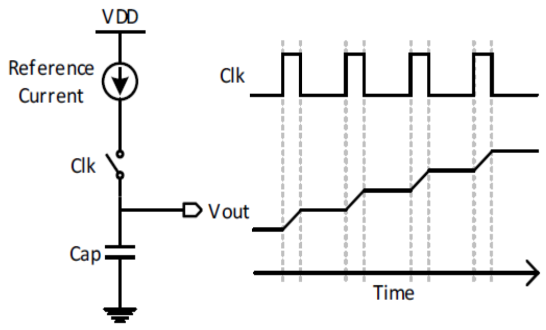


Figure 2. 3 *Trigger Circuit and Voltage build up across the capacitor*

Voltage accumulated across the capacitor during every positive half of ‘Clk’ signal shown above is:

$$\Delta V = \frac{I_{ref} \times T_{positive}}{Cap}$$

Besides, there are two important parameters of charge accumulation-based trigger circuit:

1. **Charge Accumulation:** Every time victim wire toggles, voltage across the capacitor increases by some amount. As soon as the voltage is above a predefined threshold voltage, trigger is activated, and payload is deployed. The time taken to fully activate the trigger circuit is knows as **Trigger Time**.
2. **Charge Leakage:** Once the victim wire stops toggling, the charge across the capacitor slowly leaks away. Eventually, the voltage across the capacitor falls below the threshold. The time taken for the trigger to deactivate once the victim wire stops toggling is called as **Retention Time**.

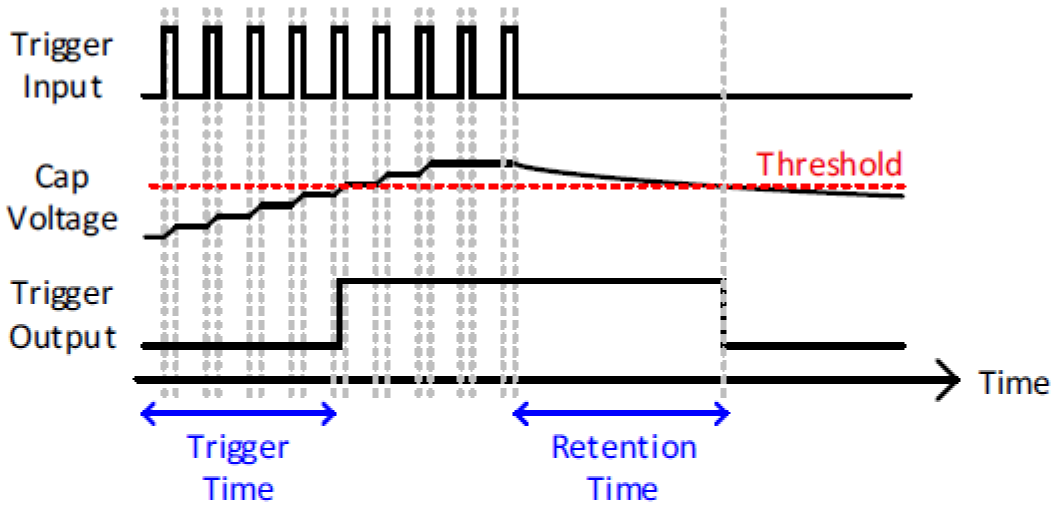


Figure 2. 4 *Trigger Time and Retention Time*

The trigger circuit has to be designed in a way that charge accumulation is stronger than the charge leakage. Otherwise, whatever charge will be accumulated during the positive half, it will be leaked way during the negative half of ‘Clk’.

Besides, the trigger circuit should be stealthy, and it should not be activated during the normal operation of circuit. The trigger circuit mentioned above is a Single-Stage Trigger Circuit as it only takes one victim wire as the trigger input. This has a higher probability of getting triggered during the normal operation of the circuit or during the testing phase , known as False Trigger Activation. To prevent this, we can combine multiple single-stage trigger circuits which has mutually exclusive victim wires as trigger inputs. This would make the multi-stage trigger stealthier and more controllable.

2.4.2 Multi-Stage Trigger Circuits

Below are some examples of Multi-Stage Trigger Circuits[2] which is a logical combination of two or more single-stage trigger outputs. To understand their operation, we will assume that the trigger circuit outputs Logic 0 when trigger is activated. For example, for OR based trigger, final trigger is activated when both A and B triggers are activated. In a similar fashion, we can use different logic combinations to make complex trigger circuits.

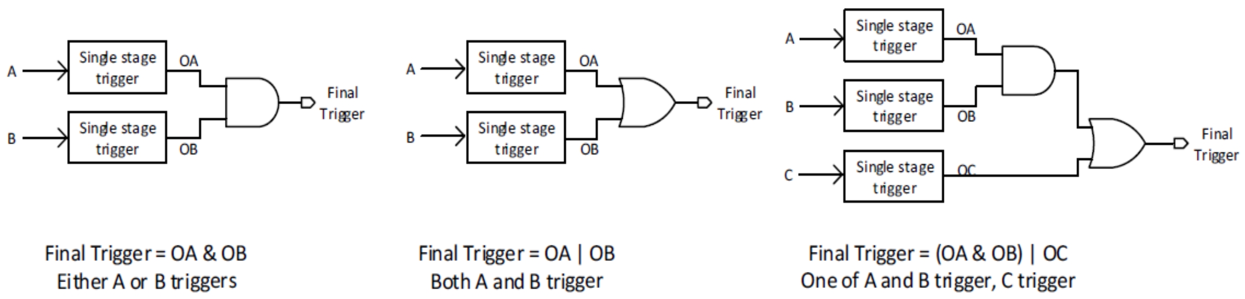


Figure 2. 5 *Examples of Multi-Stage Trigger Circuits*

Trigger Input:

Victim wires serving as a trigger input has to be the ones which has a lower activity rate during the normal operation of the circuit. Such wires are chosen by running several benchmark programs keeping in mind the expected workload of the processor. We also need to make sure that these low activity wires are easily controllable, and we can increase their toggle rate by running our attack programs.

In this attack[2], a divide-by-zero flag signal is used as a trigger input for single-stage attack. It is very unlikely during normal circuit operation to perform divide-by-zero operation repeatedly, while it is easy for the attacker to do so. Below is a pseudo program for it:

```
{r0 is a non-zero register but reads as zero in user mode}
Initialize SR[0]=0                {initialize to user mode}
while Attack_Success==0 do
  $i \leftarrow 0$
  while $i < 500$ do
    $z \leftarrow 1/0$
    $i \leftarrow i + 1$
  end while
  if $read(special\ register\ r0) \neq 0$ then
    $Attack\_Success \leftarrow 1$
  end if
end while
```

Figure 2. 6 *Attack program for a single-stage attack*

2.4.3 Payload Circuit

In this attack[2], payload is to deploy a privilege escalation of a user mode process. This can be achieved by changing one bit of a Supervision register, SR[0] in the attacked processor. Logic 0 denotes user mode and Logic 1 corresponds to supervisor mode. Thus, as a payload we change the value of this bit from logic 0 to logic 1 and give a user mode process full control over the processor.

Below is the modification we make to the reset signal path to the SR[0] flip-flop. This is an active-low reset, so the output would be logic 1 if reset is logic 0. Since the trigger output is 0 when it is activated, we put an AND gate in between the reset pin and the reset signal.

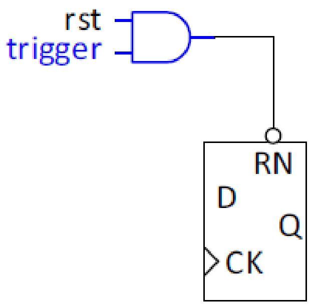


Figure 2. 7 *Payload circuit*

This payload circuit is also introduced in the unused spaces in the layout along with the trigger circuit.

2.4.4 Design Layout

Below is the Die micrograph of analog malicious hardware test chip with the inserted Trigger:

- Processor area is 2.1 mm^2
- HT circuitry occupies less than 0.08% of the chip's area

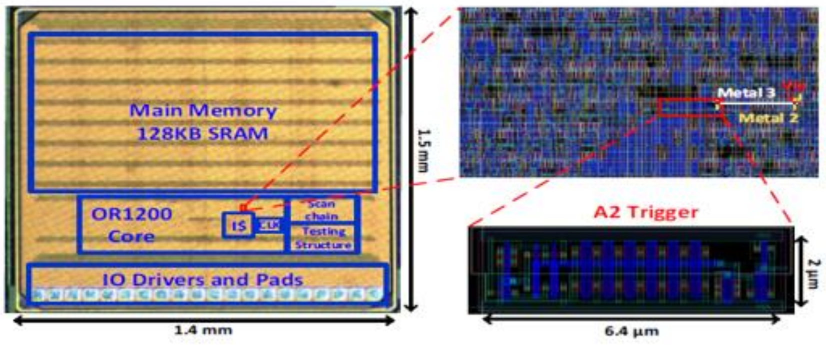


Figure 2. 8 *Die Micrograph*

Chapter 3. Possible Defenses

It is very challenging to detect Hardware Trojans due to their diversity and unpredictable process variations during fabrication. There are various techniques to detect hardware trojans, depending upon the type of impact trojans have on the original circuit. Below listed are a few techniques used to detect Trojans:

1. **IC Fingerprinting/Side Channel Analysis**: Uses side channel information like changes in power, temperature, delay profile of a chip and compare it to a golden reference chip to detect anomalous behavior.

2. **Built-In Self Authentication (BISA):** Replace the unused spaces in the placed & routed design by functional filler cells.

3. **Split IC Manufacturing**: Divide the chip into two parts. Send one part to be fabricated in a trusted fabrication house[8] and the other to a less trustworthy fabrication house.

3.1 Path Delay Fingerprinting

Determine the presence of a Hardware Trojan based upon the changes in path delay[6] of various paths in the design. There are a number of delay paths in the chip. This technique proposes that even if the inserted trojan is small compared to the rest of the chip, it has a significant impact on path delay and it can be detected. The testing procedure comprises of three steps:

1. **Path Delay gathering of Nominal chips:** Several chips are selected, and a high number of input test patterns are applied to it. The delay information of various paths is calculated. Then, the chips are reverse engineered by chip delayering and scanning them with a scanning electron microscope to ensure that the chips are genuine.

2. **Fingerprint generation:** By gathering the path delay information, a series of delay fingerprints are generated.

3. **Trojan detection:** Similar test patterns are applied to the rest of the chips and in a similar fashion, their delay fingerprints are obtained. These fingerprints are then compared to the delay fingerprints[6] of genuine chips. If there is a difference above a pre-defined threshold, the chip is labeled as malicious.

Below is one example[6] of a hardware trojan which increases the path delay of the mentioned path in the chip:

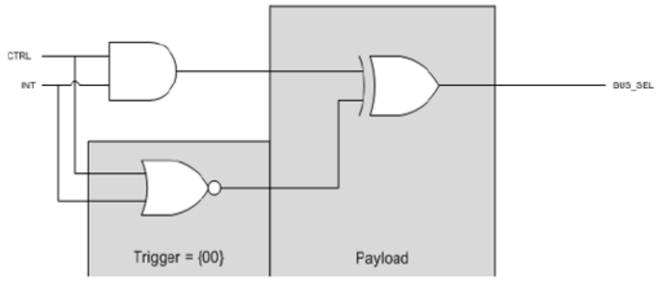


Figure 3. 1 *Trojan Circuit Example*

In the above example, ideally the BUS_SEL signal was determined by Logical AND of CTRL & INT signal. But due to the inserted trojan circuitry, the path delay increases. The trigger actively monitors the control signal and instruction interrupt signal. When both the CTRL & INT signals are logic 0, payload is activated and BUS_SEL signal gets reversed. This leads to the selection of wrong signal for inner computations.

3.2 Built-In Self Authentication (BISA)

A lot of fabrication time attacks leverage the empty spaces in the layout of a chip which are filled with dummy logic, also called as non-functional filler cells. Replacing these filler cells do not impact the functionality of the design and thus, provide a room for the attackers to insert the trojan circuitry.

This is a novel technique[9] to prevent hardware trojan insertion. The idea is to replace all the unused spaces in the layout of the design with functional filler cells. These filler cells, known as BISA cells are connected to each other to form a combinational circuit. This circuit is not connected to the original circuit and they never work simultaneously. Several input patterns are applied to this combinational circuit to obtain a digital signature. If any of the BISA cells are replaced by the attacker, the digital signature changes and hence, we can detect the presence of a trojan.

3.2.1 BISA Structure

BISA circuitry consists of three parts:

1. **BISA Circuit Under Test:** After the unused spaces are filled with BISA cells, these BISA cells are connected to each other to form a combinational logic. Instead of connecting all the cells together as one big combinational circuit, several small combinational circuits are formed, known as BISA blocks. This is done to increase the stuck-at-fault test coverage. Each BISA block operates independently.

2. **Test Pattern Generator (TPG):** It generates a number of N-bit test patterns which are shared by all the M BISA blocks. Here, it is implemented as a Linear Feedback Shift Register.
3. **Output Response Analyzer (ORA):** Each BISA block outputs a 1-bit response and thus, M BISA blocks generate a M-bit response. This M-bit response acts as an input to the ORA and ORA generates a digital signature.

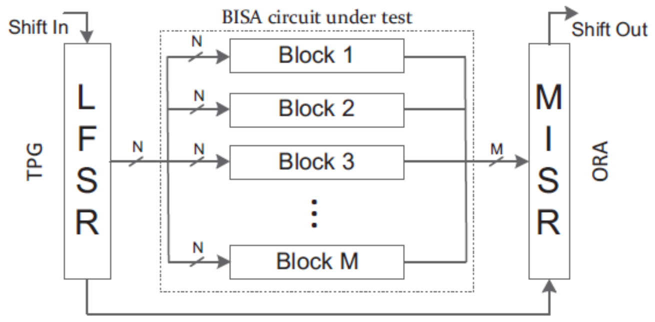
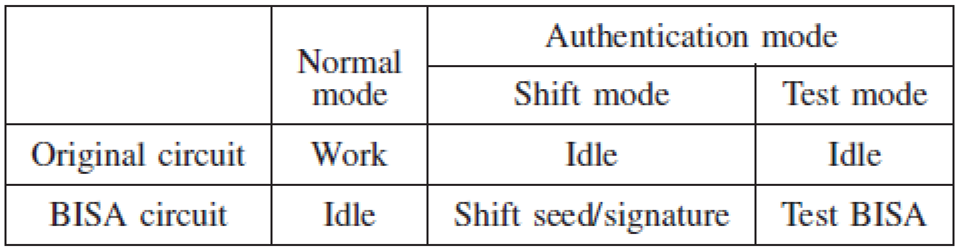


Figure 3. 2 *BISA Structure*

3.2.2 Operating Modes in a design with BISA
There are two operating modes[9] present in a design with BISA.



	Normal mode	Authentication mode	
		Shift mode	Test mode
Original circuit	Work	Idle	Idle
BISA circuit	Idle	Shift seed/signature	Test BISA

Figure 3. 3 *Operating modes*

Normal Mode: In the normal operating mode, BISA circuit is completely shutdown by disabling clocks to LFSR and MISR. The original circuit works normally as it should and BISA do not affect its functionality**.**

Authentication Mode: It comprises of two different modes:

- ***Test Mode*****:** LFSR generates N-bit test pattern which are shared by M BISA blocks at every clock cycle. At the same time, each BISA block outputs a 1-bit response. Thus, the MISR receives M bits from M blocks. We apply the test patterns for a certain number of clock cycles. After applying sufficient number of test patterns, BISA circuit can be fully tested and BISA stops. The vector stored in the MISR is the obtained digital signature. We compare the obtained signature to the signature from the simulations. If they are different, we can deduce the presence of a trojan.

- ***Shift Mode*****:** All registers in LFSR and MISR are connected in series in the shift mode. This is done so that the signature in the MISR can be shifted out and new seed for LFSR can be shifted in at the same time.

3.2.3 BISA Insertion Flow

There are some steps[9] that need to be followed for inserting BISA circuitry into the design:

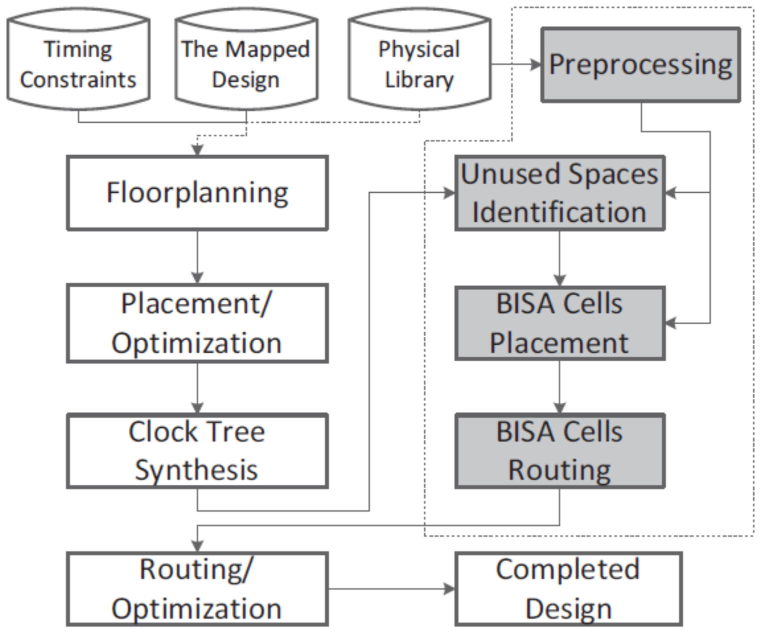


Figure 3. 4 *BISA Insertion Flow*

1. **Preprocessing:** We need to get the location of each standard cell in the design and also their length & width. This will help determining the coordinates of four corners of each standard cell.
2. **Identifying unused spaces:** A physical design tool like Synopsys IC compiler can generate a DEF file (.def) that contains the coordinates of all placed standard cells. Once we have the coordinates of the four corners of standard cells, we can obtain the locations of unused spaces in the layout.
3. **BISA Cells Placement:** A greedy algorithm is used for inserting BISA cells in the layout. We start with the largest cell in the library and try to insert it in the unused spaces. If any unused space is smaller than the size of biggest cell, we move to next biggest cell and try to fit it in. Similarly, we keep on moving towards the smallest cell available in the library until we fill all the unused spaces in the layout. Since, we insert even the smallest cell available in the library, there won’t be any room left for inserting any additional circuitry because inserted trojan has to be standard cell compatible.

4. **BISA Cells Routing:** Once all the BISA cells are placed, we need to connect them together to form BISA blocks. We try to keep the minimum number of gates in every BISA block to improve the test coverage.

Below figure depicts BISA cells insertion and placement[9]:

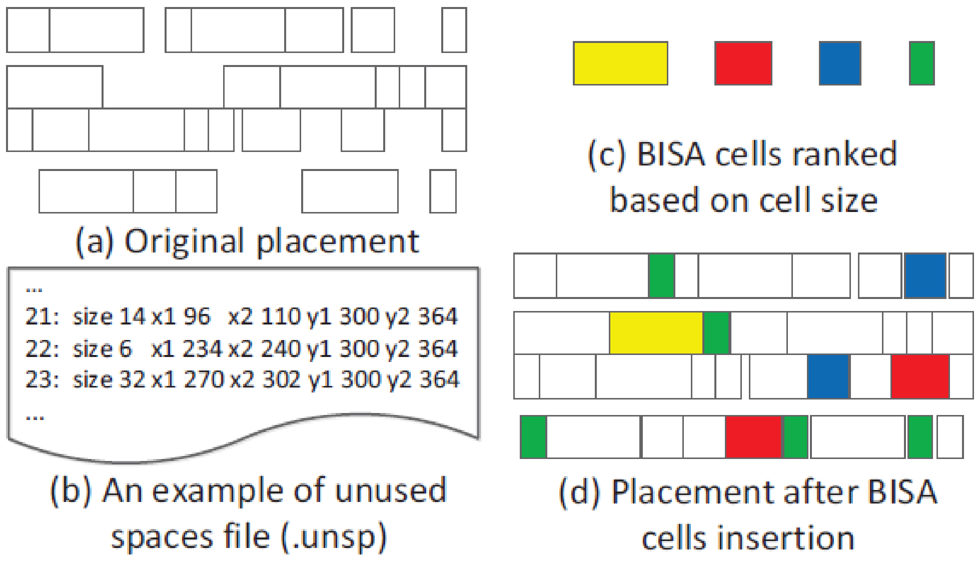
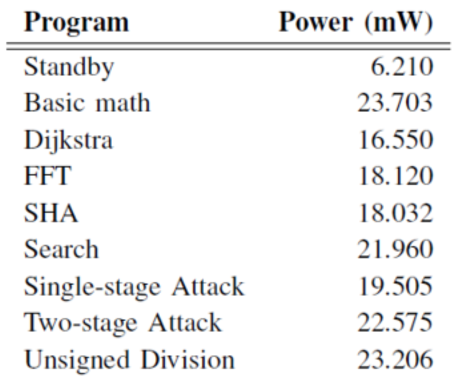

(a) Original placement
...
21: size 14 x1 96 x2 110 y1 300 y2 364
22: size 6 x1 234 x2 240 y1 300 y2 364
23: size 32 x1 270 x2 302 y1 300 y2 364
...
(b) An example of unused spaces file (.unsp)
(c) BISA cells ranked based on cell size
(d) Placement after BISA cells insertion

Figure 3. 5 *Cells Insertion and Placement*

Chapter 4. Can these Defenses detect Example Attack?

4.1 Side Channel Information (Power/Temperature)

The trojan circuitry used in our Example Attack is very small and its impact on the power consumption of the entire design is buried within the normal fluctuations in the power consumption of chip. This can be seen from the below table which depicts the power consumption of chip when certain benchmark programs are run on it. As you can see, the power consumption when the Single-stage and Two-stage is activated is very similar to when other benchmark programs are run. Such a small amount can be attributed to the normal fluctuations in the power consumption.



Program	Power (mW)
Standby	6.210
Basic math	23.703
Dijkstra	16.550
FFT	18.120
SHA	18.032
Search	21.960
Single-stage Attack	19.505
Two-stage Attack	22.575
Unsigned Division	23.206

Table 4. 1 *Chip Power Consumption*

Besides, for detection based on temperature tracking, assumption is that the temperature of the chip rises dramatically when the trojan is activated. However, in case of Example Attack, the charge in the capacitor is gradually accumulated and payload lasts for a short duration of time. Thus, rate of thermal variation is very low and cannot be detected by methods looking for a dramatic change in temperature profile of the chip.

4.2 Path Delay Fingerprinting

Because the trigger circuit in our Example Attack is small, it does not load the victim wire a lot. Thus, the timing perturbation due to the trojan insertion is masked by the noise present in the delay path.

4.3 Split IC Manufacturing

This technique is based upon dividing the chip in two parts where one part is sent to a trusted fabrication house and other to a less trustworthy fabrication house. Our Example Attack relies upon the low activity wires and a bit value in the security critical register which is used as a trigger input and victim flip-flop respectively. Thus, we can send the security critical registers and a bunch of wires including these low activity wires to be fabricated in a trusted fabrication house. Though, this would cause a certain cost overhead, but it increases the probability of protecting the chip against such attacks.

4.4 BISA

Our Example Attack is based upon a primary assumption that there will be some unused spaces in the design layout which are filled with dummy logic(known as non-functional filler cells) and removing those won't affect the functionality of the design. BISA, however, replace these unused spaces with BISA cells which are connected together to form a combinational circuit. Replacing any of these BISA cells will change the output signature of the BISA circuit and this indicates the presence of a Trojan in the chip. While this technique might prevent the Example Attack, it does have some *limitations*:

1. Significant area overhead due to the routing of BISA cells
2. Static leakage through BISA cells in the Idle mode

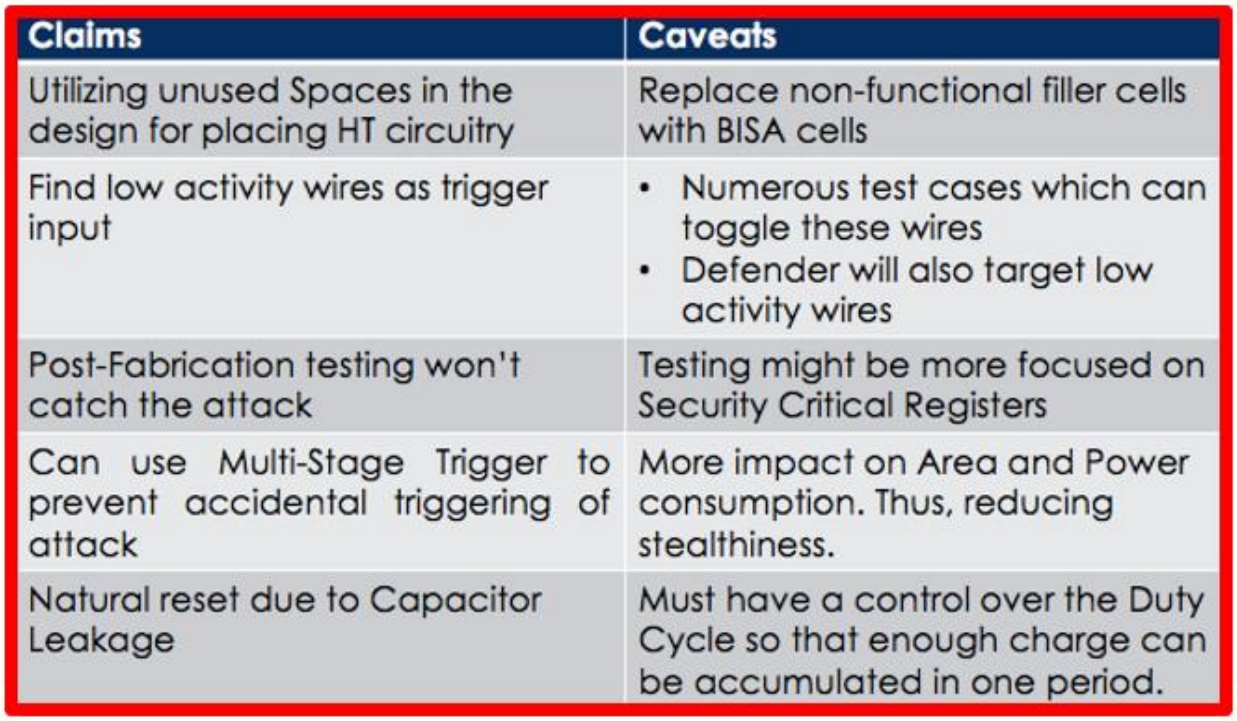
Chapter 5. Example Attack Review

The Example Attack described in Chapter 2 have some strengths along with some shortcomings which are outlined below.

5.1 Strengths

- Trojan circuitry is small, consisting of a few transistors and capacitors wrapped in a single gate
- Trigger circuit requires an unlikely sequence to get activated, which very rarely happens during normal operation or testing
- Hidden from Visual/Side-Channel inspection
- Implemented in Analog Domain so it can evade the digital verification steps
- Capacitors can naturally reset due to leakage

5.2 Shortcomings



Claims	Caveats
Utilizing unused Spaces in the design for placing HT circuitry	Replace non-functional filler cells with BISA cells
Find low activity wires as trigger input	• Numerous test cases which can toggle these wires • Defender will also target low activity wires
Post-Fabrication testing won't catch the attack	Testing might be more focused on Security Critical Registers
Can use Multi-Stage Trigger to prevent accidental triggering of attack	More impact on Area and Power consumption. Thus, reducing stealthiness.
Natural reset due to Capacitor Leakage	Must have a control over the Duty Cycle so that enough charge can be accumulated in one period.

Table 5. 1 *Shortcomings*

Chapter 6. Conclusion

- While the Example Attack is good in some ways, it is based upon some assumptions which might not hold.
- There is a diverse set of HT’s presents nowadays and there is no technique which can detect most or all of the HT attacks. We need a defense that has maximum HT detection coverage. We can do so by identifying the common behavior of most of the HTs and developing a defense to detect that common behavior.

References

1. S. S. Technology. (2012, Oct.) Why node shrinks are no longer offsetting equipment costs. [Online]. Available: http://electroiq.com/blog/2012/10/why-node-shrinks-are-no-longer-offsetting-equipment-costs/
2. Kaiyuan Yang, Matthew Hicks, Qing Dong, Todd Austin, Dennis Sylvester "*A2: Analog Malicious Hardware*" 2016 IEEE Symposium on Security and Privacy (SP)
3. G. T. Becker F. Regazzoni C. Paar W. P. Burleson "Stealthy Dopant-level Hardware Trojans" International Conference on Cryptographic Hardware and Embedded Systems ser. CHES pp. 197-214 2013.
4. R. Kumar P. Jovanovic W. Burleson I. Polian "Parametric Trojans for Fault-Injection Attacks on Cryptographic Hardware" Workshop on Fault Diagnosis and Tolerance in Cryptography ser. FDT pp. 18-28 2014.
5. D. Agrawal S. Baktir D. Karakoyunlu P. Rohatgi B. Sunar "Trojan Detection Using IC Fingerprinting" Symposium on Security and Privacy ser. S&amp;P pp. 296-310 2007.
6. Y. Jin Y. Makris "Hardware Trojan Detection Using Path Delay Fingerprint" in Hardware-Oriented Security and Trust ser. HOST Washington DC:IEEE Computer Society pp. 51-57 2008.
7. S. Kelly X. Zhang M. Tehranipoor A. Ferraiuolo "Detecting Hardware Trojans Using On-chip Sensors in an ASIC Design" Journal of Electronic Testing vol. 31 no. 1 pp. 11-26 Feb. 2015.
8. K. Vaidyanathan B. Das E. Sumbul R. Liu L. Pileggi "Building trusted ICs using split fabrication" International Symposium on Hardware-Oriented Security and Trust ser. HOST pp. 1-6 2014.
9. K. Xiao D. Forte M. Tehranipoor "A novel built-in self-authentication technique to prevent inserting hardware trojans" Computer-Aided Design of Integrated Circuits and Systems IEEE Transactions vol. 33 no. 12 pp. 1778-1791 Dec 2014.