Trojan-D2: Post-Layout Design and Detection of Stealthy Hardware Trojans - a RISC-V Case Study

Sajjad Parvin[𝕌], Mehran Goli[𝕌,*], Frank Sill Torres[◇], and Rolf Drechsler[𝕌,*]
Institute of Computer Science, University of Bremen, Bremen, Germany[𝕌]
Cyber-Physical Systems, German Research Centre for Artificial Intelligence, DFKI GmbH, Bremen, Germany[*]
Institute for the Protection of Maritime Infrastructures, German Aerospace Center, Bremerhaven, Germany[◇]
{parvin,mehran,drechsler}@uni-bremen.de,frank.silltorres@dlr.de

ABSTRACT

With the exponential increase in the popularity of the RISC-V ecosystem, the security of this platform must be re-evaluated especially for mission-critical and IoT devices. Besides, the insertion of a *Hardware Trojan (HT)* into a chip after the in-house mask design is outsourced to a chip manufacturer abroad for fabrication is a significant source of concern. Though abundant HT detection methods have been investigated based on side-channel analysis, physical measurements, and functional testing to overcome this problem, there exists stealthy HTs that can hide from detection. This is due to the small overhead of such HTs compared to the whole circuit.

In this work, we propose several novel HTs that can be placed into a RISC-V core's post-layout in an untrusted manufacturing environment. Next, we propose a non-invasive analytical method based on contactless optical probing to detect any stealthy HTs. Finally, we propose an open-source library of HTs that can be used to be placed into a processor unit in the post-layout phase. All the designs in this work are done using a commercial 28nm technology.

KEYWORDS

Optical Probing, Hardware Trojan, LLSI, Hardware Secuirty

ACM Reference Format:

Sajjad Parvin[𝕌], Mehran Goli[𝕌,*], Frank Sill Torres[◇], and Rolf Drechsler[𝕌,*]. 2023. Trojan-D2: Post-Layout Design and Detection of Stealthy Hardware Trojans - a RISC-V Case Study. In *28th Asia and South Pacific Design Automation Conference (ASPDAC '23), January 16–19, 2023, Tokyo, Japan.* ACM, New York, NY, USA, 7 pages.

1 INTRODUCTION

Integrated Circuits (ICs) has a dominant existence in every aspect of our lives, ranging from IoT devices to mission-critical applications. As the ICs become ubiquitous, security threats against ICs must be deeply scrutinized. A security breach in ICs results in catastrophic consequences. For example, an adversary can access the backdoor of ICs of autonomous cars and disrupt the speed control systems such as acceleration and brake system, to cause mayhem on the streets. The security breach in ICs is more significant in microprocessors rather than other components in a device [7, 17]. The main reason is that microprocessors are the central unit responsible for controlling and communicating with every other blocks in a system. Especially, with the rise of open-source Instruction Set Architecture (ISA) RISC-V, the security of such microprocessors must be well investigated. The openness of ISA provides attackers with more details behind the scenes. It means that even during the design process different aspects of the system can be more easily discovered and later altered by adversaries as malice.

In order to show the vulnerability of a chip to a backdoor in a real-world scenario, in 2013 Miller and Valasek [18] were able to access a Jeep Cherokee control unit remotely. In their attack scenario, these two hackers were sitting in a home miles away, turned on the music remotely, activated the windshield wipers and washers, and even killed the Jeep on a highway while it was going 70 mph. In their attack scenario, they attacked the ECU of the car and took advantage of the existence vulnerabilities in the CAN bus. Another example of such vulnerabilities was found in BMW's ConnectedDrive module. This module is responsible for sending telemetry data to the manufacturer, however, attackers were able to find vulnerabilities in the device which led to opening the cars for unauthorized drivers [5]. Hence, providing security to processors must be considered, otherwise, it will result in catastrophic consequences.

Though, a chip can be designed to be secured against any malicious attack, there exists a possibility of the insertion of a Hardware Trojan (HT) in the design-to-supply chain of chips to act as a backdoor for an adversary. The source of HT insertion in the design-to-supply chain stems from using open-source IPs in the design house, outsourcing in-house design to a foundry abroad to fabricate the design, and even in the shipping stage, a malicious counterfeit chip can be replaced with the real chip. In literature, it is shown that HT insertion in the in-house design stage can be detected [8] as CAD tools complain about the added circuitry whether integrated into IP or by a rogue engineer in the design house. Moreover, in the shipping stage of the design-to-supply chain, an attacker can reverse engineer the chip and replace it with counterfeit ones [30]. However, at this stage, the attacker's capability is very limited. Since reverse engineering, the whole chip can be a daunting task to add HT to chips. Besides, some defense mechanisms against attacks in the shipping stage are proposed, such as anti-tampering packaging and obfuscation against Side-Channel Analysis (SCA) [2]. Hence, the most HT insertion-prone stage is the untrusted foundry abroad. A rogue engineer or team of engineers can modify the GDS-II file of the design and insert HT into the chip, fabricate it, and ship the malicious chips back to the end-users.

In the fabrication stage attack, an untrusted foundry has access to the mask files, hence they can remove/add circuits to the layout of the design or they can change the fabrication process parameters such as doping alteration [3, 27] to insert a malicious functionality in the design, e.g. destruction, information leakage, etc. Moreover, this post-layout insertion of HT must have a small overhead in comparison to the whole circuit, to avoid being detected using SCA. As an example of such low overhead HT, [34] proposed an analog HT that upon applying a specific input pattern, a capacitor is charged gradually, and when the charge is reached a certain value, a malicious functionality is triggered. HTs inserted during the fabrication stage are tiny and stealthy which results in the impotence of current HT detection methods (SCA, testing, and physical measurement).

In this work, we focus on modeling and detection of hard-to-detect Layout-Level Digital Hardware Trojans (LDTs) that cannot be detected by conventional methods. We propose an analytical approach based on Laser Logic State Imaging (LLSI) to detect such LDTs and modifications in finalized GDS-II file. We demonstrate that by using LLSI, we do not need a golden chip, and only a golden design is required. In summary, the main contributions of this paper are as follows:

- developing and insertion of three new LDTs into different blocks of the post-layout of an in-house designed 32-bit integer-based instruction set single-cycle RISC-V core (RV32IM),
- proposing a non-invasive analytical approach based on LLSI to detect such LDTs, and
- creating an open-source layout library of LDTs in 28nm [1].

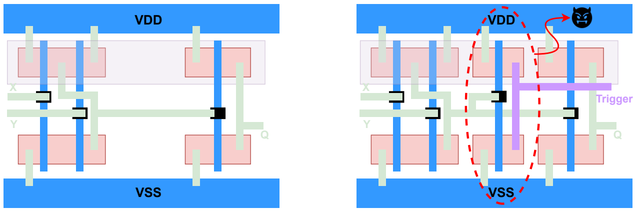
2 RELATED WORKS

HT can be inserted and detected in various levels of abstraction in the digital design flow, i.e, from transistor-level [3, 34] to RTL [25, 26]. In the following, we discuss methods that are related to our proposed approach.

In a design house, it is possible that the design team utilizes a third-party IP that may contain a malicious part (an HT is inserted in the IP's RTL source code/or netlist/or layout). Though this type of HT insertion is unchallenging for the design house team as the detection of such HTs is well-studied. There exist several methods to expose such HTs, e.g, analyzing the HDL source code of soft-IPs for HT [1, 10, 35], using formal verification methods to verify the IPs do not include unintended functionalities [9, 16, 32], and the use of design-for-trust techniques are proposed to identify malicious circuits [15, 23]. Another HT insertion scenario is to have a rogue engineer in the design house insert the HT in the design. However, it is shown that the EDA tool will complain about such modification in the final stages of the pre-silicon phase [8].

On the other hand, after outsourcing the design to an untrusted foundry abroad, an HT can be inserted into the mask design, which can be stealthy and triggers in a rare condition with the intention of information leakage or Denial-of-Service (DoS). At the fabrication stage attack, Post-layout Hardware Trojan (PHT)s are designed to be infinitesimal whether by inserting a small circuitry [4, 14, 33, 34], or modifying transistors' fabrication parameters [3, 6, 13, 27] that can go undetected using conventional detection methods, i.e. SCA, testing, etc. It must be noted that PHT includes both digital circuit HT (LDT) and analog circuit HT namely Layout-level Analog Trojan (LAT).

Due to the negligible area overhead and power consumption of PHTs in comparison to the entire chip, detecting PHTs using SCA (e.g., power analysis, and EM analysis) is impossible [14, 34]. Especially, when the PHTs are inserted by modifying transistors fabrication process parameters (i.e., change of transistors' dopant type (LAT type HT), nothing extra is added to the chip to appear in the SCA profile [3]. Besides, to detect such phantom-like PHTs using SCA, having a golden chip is a must. However, a golden chip might not be available for the design.



(a) HT free layout design. **(b) Adding a HT into the layout.**

Figure 1: Portion of a chip layout with and without HT.

The other method to detect PHTs (that are inserted into the design by modifying the transistors fabrication process) is to reverse engineer a chip by studying the de-layered model of the chip using an SEM microscope [29]. However, this technique is invasive and destructive. Moreover, de-layering a chip requires huge human labor and uses chemicals that can cause corrosion on the chip. The corrosion of the chip might result in the removal of some layers (e.g., removal of a modified part of the chip). Hence, these aforementioned disadvantages show that PHT detection using de-layering is very error-prone, which result in increasing the false-negative detection.

3 HT DETECTION CHALLENGES IN LAYOUT

LDTs, e.g., HTs that leak secret assets (e.g. crypto keys) require several flip-flops plus a control unit [25], which occupies a huge area on a chip. Hence, they can be exposed using conventional SCA [2] methods such as power analysis, and EM analysis. However, as shown in Fig. 1, if a small LDT (that causes, for instance, DoS) is inserted into the layout of a chip during its fabrication phase, the detection process might not be easy to expose. This is due to the fact that these small LDTs have negligible information leakage through the SCA or are activated in rare conditions. This makes their detection hard. On the other hand, to expose HT in a chip using SCA a golden chip is also required which might not be available for every design. The possibility of adding extra circuitry to the GDS-II file of a design in an untrusted foundry is well studied and discussed in [4]. According to [4], if the density of a chip is below 80%, it is possible to add extra circuitry to the GDS-II file of a chip without the need for re-doing the place and route. In addition, adding extra circuitry to the GDS-II file in the post-layout stage is discussed in [21, 22] and even it is called to be a trivial task for an IC expert. Hence, for an expert IC designer, it is easy to insert a small digital circuitry in a jungle of transistors, without the need to re-design the entire chip. HT insertion into the GDS-II file flow for a rogue engineer in a foundry is shown in Fig. 2. Firstly, a rogue engineer retrieves the netlist of a design from the original GDS-II file [22]. Secondly, based on the retrieved netlist, she/he tries to insert an LDT into the design. Thirdly, she/he applies engineering change order (ECO) routing. By ECO routing, the routing of the original design, is kept intact [21]. Finally, the modified GDS-II is sent for fabrication.

[1] Bounded by NDA, we can not release the library in commercial technology. Hence, the released library will be in NCSU 45nm open-source technology file.

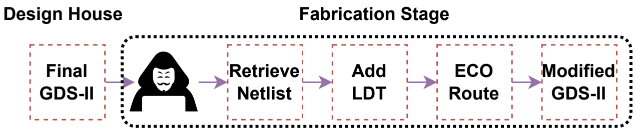


Figure 2: Flow of HT insertion in an untrusted foundry.

While the aforementioned methods can help designers to detect LDTs that occupy/consume noticeable area/power on a chip, for the case of stealthy LDTs, they come with several drawbacks that can be summarized as follows:

- HT detection using SCA is incompetent as stealthy HT can remain hidden from detection.
- Using SCA to detect stealthy HT requires golden chip which might not be available for all designs.
- If no re-routing is applied to the GDS-II file of the design after HT insertion, HT can hide from being detected using optical inspection.
- De-layering and investigation of a chip using SEM microscope is time-consuming, invasive, and expensive.

Hence, the question still remains unanswered:

How can we detect HTs with a negligible area and power overhead (as depicted in Fig. 1) that are inserted during the fabrication phase in an untrusted foundry, efficiently?

In order to detect the insertion of stealthy LDT in the design during the fabrication phase, we propose a novel approach based on Optical Probing (OP), more specifically, the LLSI technique. The proposed approach is a non-invasive HT detection that does not require a golden chip. It works based on the comparison between LLSI simulation of a golden layout and LLSI analysis of the fabricated chip.

4 HT DETECTION USING OPTICAL PROBING

In this section, we will discuss the optical probing setup and fundamentals of OP resolution and technology. Next, we formulate the OP of a transistor and logic gate to be used in our LLSI simulation.

4.1 Optical Probing Setup and Resolution

The idea of OP comes from the fact that silicon is transparent to light in Near-InfraRed (NIR) spectrum. Hence, it is possible to probe/image the backside of a chip. OP utilizes a focused laser beam pointed to the backside of the chip. This light traverse through various regions of a MOSFET, and based on the electrical field present at the junction of the MOSFET, the reflection and refraction coefficient of the laser beam gets modulated (this modulation is due to the presence of an electrical field in the transistors). Then, a detector collects all the reflected light. This reflected light at the detector corresponds to the voltage present at the terminal of the MOSFET. The setup for the OP is shown in Fig. 3, where a laser is pointed to the back side of a chip in the flip-chip package.

In general, there exist two methods to probe an IC through the backside using OP; 1) Electro-Optical Probing (EOP) where a focused laser is parked on a spot of an IC to read out the waveform present at that spot, and 2) Electro-Optical Frequency Mapping (EOFM) where a laser scans the IC area of interest using a galvanometric mirror to find active regions carrying signals with a specific frequency [31]. Besides, there exists another method called LLSI to retrieve the on or off state of transistors. It is an extension of EOFM where the power line of the IC is modulated with a small Sinusidal signal (amplitude: 100mV, frequency: 100KHz), and then the IC's area of interest is scanned [19]. Due to modulating the power line of an IC, all logic gates' (both sequential and combinational logic gates) active regions pop up in the LLSI analysis.

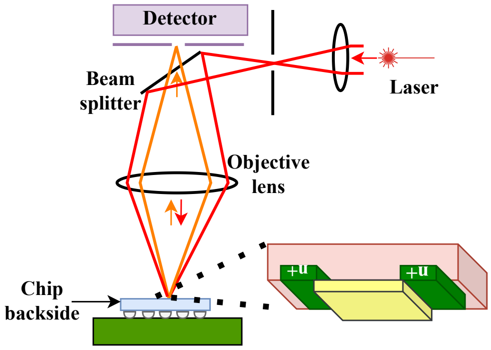


Figure 3: Optical probing setup.

4.2 Optical Resolution

There are several definitions for spatial resolution R, though the most commonly used formula for optical resolution is defined by Fourier optics and Abe's criterion [24] as $R = \frac{0.5\lambda}{NA}$ where λ, NA are light wavelength, and numerical aperture, respectively. Optical resolution is defined as the minimum distance between two objects that an imaging system can distinguish. The laser spot's intensity can be expressed as a Gaussian distribution [24] as follow:

$$p(r) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(r)^2}{2\sigma^2}} \tag{1}$$

where r is the distance from the center of laser spot, and σ is the standard deviation which can be calculated as $\sigma = \frac{0.37\lambda}{NA}$ [24].

The optical resolution (R) is only related to the λ and NA parameters. Hence, an improvement can be obtained by either reducing the λ or increasing the NA parameters. In this work, for all simulations, we used laser wavelength and NA of 1300nm and 3.5, respectively.

4.3 LDT Detection Using LLSI

Performing LLSI requires modulation of the power line of a chip with a 100mV signal at 100kHz. Hence, all the transistors in the chip will have a small sinusoidal signal on them. Then by scanning the chip area using OP and comparing the LLSI image of the golden design, we can spot the inserted LDT in the chip as an extra pattern on the chip. To perform the LLSI analysis on the golden layout, we assume that we have the reflection of PMOS and NMOS transistors' regions used in the design when the power line of the IC is modulated with a 100mV signal at 100kHz, namely Reflection Caliber Value (RCV) [20]. Modulating the power line of chip results in having a signal on all the transistors in the design. The RCV of a region of a MOSFET is expressed as

$$RCV = V \times K \times \beta \times P_L \int_0^{2\pi} \int_0^{r_{spot}} p(r) \times A(r,\theta)\, dr d\theta \tag{2}$$

where V, P_L, A, and $p(r)$ are voltage at the terminal of a transistor's active region, laser's power, area of the active region under the laser spot, and laser's intensity distribution shown in (1). The parameters K, and β are transistor's fabrication related parameters [20]. In addition,

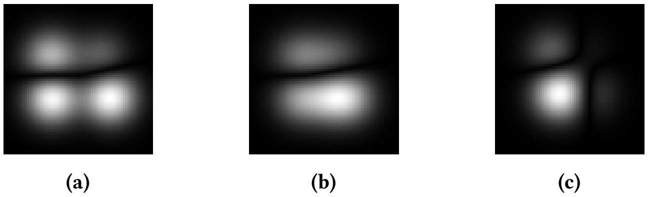


Figure 4: (a) LLSI simulation on golden layout, (b) LLSI analysis on fabricated chip, (c) difference between LLSI analysis of the fabricated IC and the golden layout.

this formulation can be expanded for each active region of a transistor. The RCV of a transistor is the summation of RCV for each region (R) of the transistor (drain (D), source (S), gate (G)) which can be described as follow:

$$RCV_{FET} = \sum_{\forall R \in trans.\{D,S,G\}} RCV_R. \tag{3}$$

To perform LLSI on a golden layout, we need to further expand the RCV_{FET} to RCV_{CELL} as each gate cell consists of several transistors. The RCV_{Cell} parameter can be expressed as follow:

$$RCV_{Cell} = \sum_{\forall t \in Cell} RCV_{FET_t}, \tag{4}$$

where the t, and RCV_{FET} represent each transistor in each gate cell, and reflection caliber value of a transistor as shown in (3), respectively.

Furthermore, based on golden layout's netlist, we can find out which transistor in which logic gate has positive voltage on its terminal based on the applied input pattern. Using the RCV value of the transistors, and consequently, the RCV of each logic gate, as discussed previously, we perform convolution between the RCV of the active region of the logic gates and the golden layout image of size i, and j (M) of the area of interest. Layout matrix M and the LLSI analysis image can be formulated as follow:

$$M_{(i,j)} = \begin{cases} 1 & \textit{if there is a transistor at position i \& j} \\ 0 & \textit{otherwise} \end{cases} \tag{5}$$

$$LLSI_{image}[i,j] = \sum_{k=1}^{m} \sum_{l=1}^{n} M(i+k-1, j+l-1) RCV_{Cell}(k,l). \tag{6}$$

By comparing LLSI images of the IC and the golden layout, we can conclude the existence of LDT. Hence, if an additional circuit is added to the IC during the fabrication phase, there will be more light reflection from the chip. By simply differentiating between two images, the LDT can be exposed.

Going back to our motivating example in Fig. 1. Assume that the circuit shown in Fig. 1a is our golden layout design, and the circuit shown in Fig. 1b, is the fabricated one. The security scenario here is that a rogue engineer in an untrusted foundry inserts an additional INVERTER gate into our design. By performing LLSI analysis on these two circuits, we obtain two separated images. The LLSI analysis is shown in Fig. 4. By taking a difference between these two images (illustrated in Fig. 4c), the inserted LDT and its location on the chip are exposed.

The difference originated from the amount of light reflection between the fabricated chip and the golden layout's LLSI images.

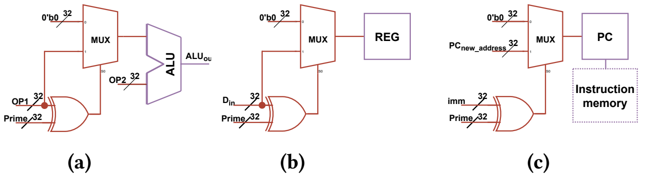


Figure 5: (a) TRJ1 is placed in the ALU, (b) TRJ2 is placed in register, (c) TRJ3 is in program counter.

5 LDT DESIGN AND DETECTION

In order to evaluate the insertion of LDT in a real-world design and to show the effectiveness of our approach in the detection of such HTs, we took advantage of an in-house designed 32-bit single-cycle based integer instruction set RISC-V core (RV32IM). The synthesis, and place and route processes were done in a commercial CMOS Bulk 28nm technology. Our RV32IM core is designed to be a System on a Chip (SoC). The designed SoC is equipped with 1MB of data memory (DSRAM) and instruction memory (ISRAM). For interfacing with the chip, we utilized parallel to serial (P2S) and serial to parallel (S2P) modules. The design specification of the SoC is shown in Table 1. The design is done using Cadence tools and verified using the Mentor Calibre tool.

As discussed in [21, 22], it is a trivial task for an adept IC designer to insert additional circuitry into the post-layout GDS-II file of a design. We inserted three LDTs, shown in Fig. 5 into the layout of RV32IM core of our SoC as shown in Fig. 6. The inserted LDTs into the design are discussed in detail in the following subsection.

5.1 ALU Unit Trojan Design (TRJ1)

In the first scenario, we place the LDT shown in Fig. 5a, in the ALU unit of the RV32IM core. Upon having one of the operands (in our case **OP1**) equal to a prime number (set by the attacker), the output of the ALU becomes zero. This LDT disrupts the normal behavior of the system. The post-layout measurement of the **TRJ1** is shown in Table 1. In comparison to the entire SoC, the area overhead (0.03%), and power consumption (0.19%) of this LDT are negligible. Hence, using conventional HT detection methods such as SCA, we cannot detect the insertion of such an LDT.

5.2 Program Counter Trojan Design (TRJ2)

In the second scenario, the designed LDT is placed in the program counter unit of the RV32IM core. Upon having an immediate jump address equivalent to a certain prime number that is set by the attacker, the program counter is set to reset. The aim of this LDT is to cause the SoC to experience DoS. As demonstrated in Table 1, the occupied area by this LDT is only 0.02% of the entire SoC. Moreover, the delay and power consumption overheads of this LDT are infinitesimal. The schematic of **TRJ2** is shown in Fig. 5b.

5.3 Register File Trojan Design (TRJ3)

In the last scenario, the LDT is placed in the register file block of the RV32IM core. This LDT is similar to the previously explained LDTs, which result in DoS for our SoC. Upon having a true result for a comparison between a prime number set by the adversary in the chip and the input data to the register files, the register data will be corrupted. The schematic diagram of the **TRJ3** is shown in Fig. 5c. Moreover, as shown in Table 1, **TRJ3** adds a very small area,

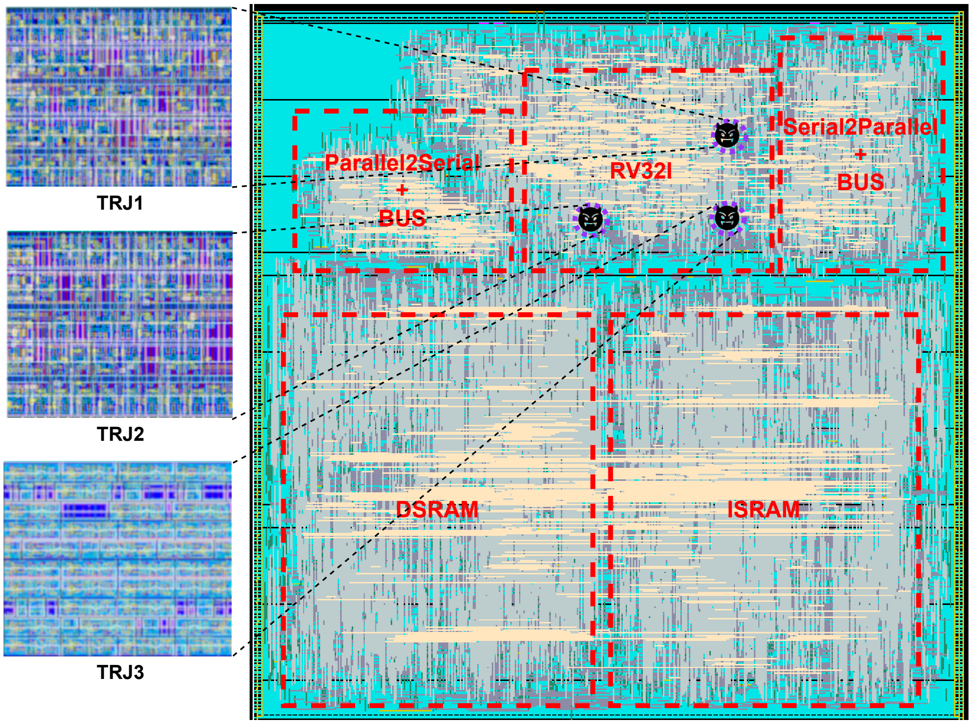


Figure 6: Post-Layout figure of RISC-V with Trojans.

power, and delay overhead to the SoC. According to Table 1, **TRJ3** has a power consumption of 4% of the total SoC (the largest power consumption among all our three HTs). The main reason for this high value is the fact that we target the entire cells of the register file to be corrupted upon activation of the **TRJ3**. However, an attacker can just manipulate a single bit of a given register cell e.g., the most significant bit. Therefore, the power consumption of **TRJ3** would be much smaller.

Please note that, in our experiment, we consider a very small block of memory for both ISRAM and DSRAM, enabling us to only introduce the proposed LDTs and show the entire image of the SoC in the paper. However, in a real system, these blocks of memories are huge. This means that the reported non-functional overhead of LDTs in Table 1, especially (**TRJ3**) can be much smaller. Thus, the detection process using conventional approaches can be impossible.

5.4 Trojan Detection

A common security analysis practice is that first all critical parts of a chip are analyzed. Based on initial results, security verification engineers decide whether more analyses are required on the non-critical parts of the chip or not [31]. Since we know the sensitive area of the design is the RV32IM core of our SoC, we omitted performing LLSI analysis on the whole SoC. Hence, to perform LLSI analysis on the design, we assume each NMOS, and PMOS have the generic reflection of -1.3, and 1 [20], respectively. Then, we apply the same input value to the RV32IM core's golden design's netlist and malicious RV32IM core's netlist. Based on the input value applied to these two cores, we know which transistors in which cell gate in these two cores are on or off. Then, we assigned a generic optical reflection value of -1.3 or 1 to active PMOSes and NMOSes, respectively. If a transistor is off, this means the transistor is not contributing to light reflection. Hence, we assign a reflection value of zero to the transistors that are off.

Table 1: Design specification of SoC, inserted LDTs, and comparison of the design specs.

Design	P (μW)	A (μm^2)	D (ps)	$\frac{P_{trj}}{P_{SoC}}$	$\frac{A_{trj}}{A_{SoC}}$	$\frac{D_{trj}}{D_{SoC}}$
SoC	101	122500	2119	-	-	-
TRJ1	0.19	31.6	132	0.19%	0.03%	6.23%
TRJ2	0.31	25.3	19	0.31%	0.02%	0.90%
TRJ3	3.90	95.9	8	3.86%	0.08%	0.38%

Next, we scan the whole region of both golden layout and malicious RV32IM cores, as explained in Section 4, Fig. 7a and Fig. 7b show the LLSI analysis of the RV32IM core without any modification, and the malicious RV32IM core, respectively. By simply taking the difference between these two LLSI analysis images, we can find the existence of extra circuitry in our design, as shown in Fig. 7c. In the case of LDT, we have extra circuits in the design. Having extra circuitry in a design means we have more reflection of our design compared to the original design. It is worth mentioning that using this technique to expose LDTs can also be used to detect HTs where a rogue engineer in the foundry removes some part of the design.

6 INTEGRATION AND DISCUSSION

In this section, we discuss the novelty and importance of modeling LDTs in ASIC design flow, the results of our proposed optical-based detection of LDTs approach as well as how to obtain optical reflection of each transistor for performing LLSI in simulation on a golden layout. Moreover, we explain solutions for the limitations of our LDT detection approach; the effect of process variation on the optical reflection, and optical probing process time. Finally, we introduce our open-source library of HT.

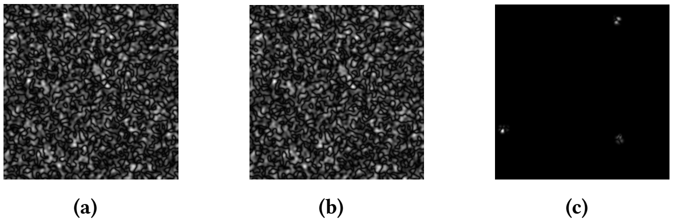


Figure 7: (a) LLSI simulation on RV32IM core's golden layout, (b) LLSI analysis on fabricated chip, (c) difference between LLSI analysis of the fabricated IC and golden layout.

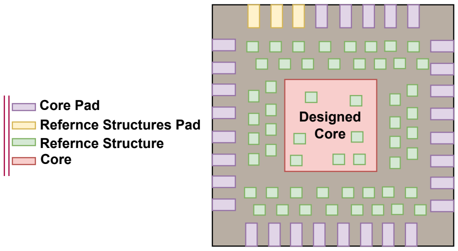


Figure 8: Reference structure distribution diagram to retrieve reference OP reflection.

6.1 Trojan Design and Detection

In this paper, we demonstrated how LDTs can be modeled and inserted into the finalized GDS-II file of the design (that can be performed in an untrusted foundry). Using our proposed OP-based HTs detection approach, in particular, LLSI to detect inserted LDTs in the finalized GDS-II file of the design. As mentioned earlier, OP to detect HT has been discussed in the literature e.g., [12, 28]. The focus of the aforementioned works is on the HTs insertion and OP detection in an FPGA. Please note that HT insertion in an FPGA is not very logical. Since the FPGA's synthesis tool might configure the FPGA differently, in each run, different bitstreams can be loaded. Hence, running the OP on an FPGA with and without HT might result in the activation of totally different areas. However, in Application-Specific Integrated Circuit (ASIC) flow, upon insertion of HT into a design, the nature of the design is kept intact. This means that the attacker only removes the filler cells in the design and replace them with extra circuitry. In other words, in the case of adding HT in an ASIC, the malicious chip and golden layout are kept almost intact, though inserting HT in FPGA, malicious FPGA and the HT free FPGA might differ greatly, in terms of layout.

Furthermore, we showed adding a small layout to a design that is placed in the empty areas of the chip can be detected using the LLSI technique. The main reason is that in the LLSI technique, the power line of the chip is modulated with a small sinusoidal signal (100mV, and 100KHz). Modulating the power line means all the standard cells connected to the power line have this sinusoidal signal on their active regions of the transistor in each logic gate. Hence, by comparing LLSI analysis of malicious chip and golden layout, the inserted LDTs are exposed, as shown in Fig. 7. In addition, to exploit the LLSI scheme to detect LDTs, there is no need for applying test cases. Even a random input can expose the LDTs as LDTs consist of several logic gates, and each logic gate contains several transistors. By having any input pattern, the modulated signal on top of the power line is presented at the transistors in the LDTs' logic gate. In other words, **there is no need for activation of LDT**.

6.2 Obtaining optical reflection to perform LLSI simulation on golden layout

In this work, we assumed that the reflection of each transistor under LLSI is information that we possess. In reality, to find such information, two approaches can be used; 1) asking the foundry to provide us with the technology details under NDA (then based on fabrication parameters of transistors, we can find the refraction and absorption of laser light [11]), 2) using a test chip fabricated with single transistors. Both of the aforementioned methods are not feasible. If a foundry is untrusted, the provided information from them might be flawed or they refuse to provide such sensitive information. Moreover, fabricating a chip with single transistors might not be feasible, due to its high cost. The other approach is to distribute single transistors of various sizes around the chip (or in empty spaces of the design) as shown in Fig. 8 that is being sent for fabrication. Then, these reference structures are used to retrieve the OP reflection of each transistor after performing OP analysis on the chip. Consequently, we can use these reference reflection values in our LLSI simulation for our golden layout.

6.3 Process Variation (PV)

As discussed in [11, 20], optical reflection is dependent on the fabrication parameters (e.g., doping, and size). Hence, PV can be a limitation to our detection approach. However, we can use the idea of distributing some reference structures around the chip, as shown in Fig. 8 to have a reference value for optical probing. In other words, these reference structures can also serve as structures to show the effect of process variation on the reflected light from OP. Since PV occurs in various regions, we can load the reflection reference values for regions with PV into the golden layout's LLSI analysis to improve the LDT detection accuracy.

Another important point that should be taken into account is that typically, in post-silicon flow, a fabricated chip goes through intensive functional testing for functional correctness and PV detection with great accuracy. We assume our fabricated chip and golden design have passed functional testing.

6.4 Time as a limitation to LLSI detection

We only performed LLSI analysis on both golden-layout and malice layout in simulation. In our LLSI analysis, the simulation results were obtained fast. However, in a real-world scenario, performing LLSI on a chip could take much longer. This is due to the relatively low SNR at each step where the laser moves over the chip when performing LLSI. It is possible to improve the timing of LLSI by increasing the steps that lasers move over the chip, staying on each spot for a shorter amount of time (i.e. decreasing SNR of the overall image). There is a need to find an optimal point for good image SNR and LLSI analysis time. The needed time for completion of LLSI analysis in the real world scenario for our RV32IM core which is approximately 60μm ×60μm is in the order of minutes to a few hours (depending on a trade-off between time and SNR of image). However, the required time for such a critical offline analysis is reasonable.

6.5 Open source HT layout library

We released our designed LDTs' GDS-II files online [2]. Furthermore, our library contains a single-cycle RISC-V core (RTL, and layout), layout and RTL code of each LDT discussed in this paper with various bit-size configurations. Moreover, this open-source HT library

[2] Link to the repository: https://github.com/Saazh/Trojan-D2

allows researchers to study the detection of such HTs using various techniques and benchmark their detection methods.

7 CONCLUSION

In this work, we designed three novel LDTs for a single-cycle 32-bit integer-based instruction set RISC-V core. These LDTs are inserted into the finalized GDS-II file of our SoC in a scenario where the layout is in the hand of a rogue engineer in an untrusted foundry. These LDTs upon triggering, result in DoS in our SoC. Moreover, we showed that these LDTs have a negligible non-functional overhead (area, power and performance) compared to the entire design. As a result, conventional HT detection methods are liable to find such small LDTs. Hence, we proposed an LLSI-based approach allowing designers to perform optical probing on the backside of a chip (LLSI analysis) to detect LDTs. Using LLSI to detect LDT does not require a golden chip (which might not be available in all cases), instead, it requires the golden layout, which is available in the design house. In addition, we made the LDTs layout open-source.

For our future work, we tape out our proposed SoC using a commercial 28nm technology. This is done for a design with and without LDTs. We are interested in evaluating, how close our detection methods can be in comparison to our simulation result. Moreover, we would like to investigate the detection of LATs in the design using the LLSI technique.

ACKNOWLEDGMENTS

The work described in this paper has been supported by the Deutsche Forschungsgemeinschaft (DFG – German Research Foundation) under the priority programme SPP 2253 – 439918011 in project DR 287/38-1.

REFERENCES

[1] Mainak Banga and Michael S. Hsiao. 2010. Trusted RTL: Trojan detection methodology in pre-silicon designs. In *2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*. 56–59. https://doi.org/10.1109/HST.2010.5513114

[2] Alex Baumgarten, Michael Steffen, Matthew Clausman, and Joseph Zambreno. 2011. A case study in hardware Trojan design and implementation. *International Journal of Information Security* 10, 1 (2011), 1–14.

[3] Georg T Becker, Francesco Regazzoni, Christof Paar, and Wayne P Burleson. 2014. Stealthy dopant-level hardware trojans: extended version. *Journal of Cryptographic Engineering* 4, 1 (2014), 19–31.

[4] Shivam Bhasin, Jean-Luc Danger, Sylvain Guilley, Xuan Thuy Ngo, and Laurent Sauvage. 2013. Hardware Trojan horses in cryptographic IP cores. In *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography*. IEEE, 15–29.

[5] Zhiqiang Cai, Aohui Wang, Wenkai Zhang, M Gruffke, and H Schweppe. 2019. 0-days & mitigations: roadways to exploit and secure connected BMW cars. *Black Hat USA* 2019 (2019), 39.

[6] Samaneh Ghandali, Georg T Becker, Daniel Holcomb, and Christof Paar. 2016. A design methodology for stealthy parametric trojans and its application to bug attacks. In *International Conference on Cryptographic Hardware and Embedded Systems*. Springer, 625–647.

[7] Mehran Goli and Rolf Drechsler. 2021. Early Validation of SoCs Security Architecture Against Timing Flows Using SystemC-based VPs. In *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. 1–8. https://doi.org/10.1109/ICCAD51958.2021.9643579

[8] Alexander Hepp and Georg Sigl. 2021. Tapeout of a RISC-V crypto chip with hardware trojans: a case-study on trojan design and pre-silicon detectability. In *Proceedings of the 18th ACM International Conference on Computing Frontiers*. 213–220.

[9] Yier Jin and Yiorgos Makris. 2013. A proof-carrying based framework for trusted microprocessor IP. In *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 824–829.

[10] Jing-Yang Jou and Chien-Nan Jimmy Liu. 1999. Coverage analysis techniques for HDL design validation. *Proc. Asia Pacific CHip Design Languages* (1999), 48–55.

[11] Ulrike Kindereit. 2009. *Investigation of laser-beam modulations induced by the operation of electronic devices.* Doctoral Thesis. Technische Universität Berlin, Fakultät IV - Elektrotechnik und Informatik.

[12] Thilo Krachenfels, Jean-Pierre Seifert, and Shahin Tajik. 2021. Trojan awakener: Detecting dormant malicious hardware using laser logic state imaging. In *Proceedings of the 5th Workshop on Attacks and Solutions in Hardware Security.* 17–27.

[13] Raghavan Kumar, Philipp Jovanovic, Wayne Burleson, and Ilia Polian. 2014. Parametric Trojans for Fault-Injection Attacks on Cryptographic Hardware. In *2014 Workshop on Fault Diagnosis and Tolerance in Cryptography*. 18–28. https://doi.org/10.1109/FDTC.2014.12

[14] Lang Lin, Markus Kasper, Tim Güneysu, Christof Paar, and Wayne Burleson. 2009. Trojan side-channels: Lightweight hardware trojans through side-channel engineering. In *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 382–395.

[15] Chen Liu, Jeyavijayan Rajendran, Chengmo Yang, and Ramesh Karri. 2013. Shielding heterogeneous MPSoCs from untrustworthy 3PIPs through security-driven task scheduling. In *2013 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*. 101–106. https://doi.org/10.1109/DFT.2013.6653590

[16] Eric Love, Yier Jin, and Yiorgos Makris. 2011. Proof-carrying hardware intellectual property: A pathway to trusted module acquisition. *IEEE Transactions on Information Forensics and Security* 7, 1 (2011), 25–40.

[17] Tao Lu. 2021. A Survey on RISC-V Security: Hardware and Architecture. https://doi.org/10.48550/ARXIV.2107.04175

[18] Charlie Miller and Chris Valasek. 2015. Remote exploitation of an unaltered passenger vehicle. *Black Hat USA* 2015, S 91 (2015).

[19] Baohua Niu, Grace Mei Ee Khoo, Yuan-Chuan Steven Chen, Fernando Chapman, Dan Bockelman, and Tom Tong. 2014. Laser Logic State Imaging (LLSI). In *ISTFA 2014*. ASM International, 65–72.

[20] Sajjad Parvin and et al. 2022. Toward Optical Probing Resistant Circuits: A Comparison of Logic Styles and Circuit Design Techniques. In *ASP-DAC*.

[21] Tiago Perez and Samuel Pagliarini. 2021. Hardware Trojan Insertion in Finalized Layouts: a Silicon Demonstration. *arXiv preprint arXiv:2112.02972* (2021).

[22] Rachel Selina Rajarathnam, Yibo Lin, Yier Jin, and David Z Pan. 2020. ReGDS: a reverse engineering framework from gdsii to gate-level netlist. In *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 154–163.

[23] Jeyavijayan JV Rajendran, Ozgur Sinanoglu, and Ramesh Karri. 2016. Building trustworthy systems using untrusted components: A high-level synthesis approach. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 24, 9 (2016), 2946–2959.

[24] Venkat Krishnan Ravikumar, Gabriel Lim, Jiann Min Chin, Kin Leong Pey, and Joel KW Yang. 2018. Understanding spatial resolution of laser voltage imaging. *Microelectronics Reliability* 88 (2018), 255–261.

[25] Hassan Salmani, Mohammad Tehranipoor, and Ramesh Karri. 2013. On design vulnerability analysis and trust benchmarks development. In *2013 IEEE 31st international conference on computer design (ICCD)*. IEEE, 471–474.

[26] Bicky Shakya, Tony He, Hassan Salmani, Domenic Forte, Swarup Bhunia, and Mark Tehranipoor. 2017. Benchmarking of hardware trojans and maliciously affected circuits. *Journal of Hardware and Systems Security* 1, 1 (2017), 85–102.

[27] Yuriy Shiyanovskii, F Wolff, Aravind Rajendran, C Papachristou, D Weyer, and W Clay. 2010. Process reliability based trojans through NBTI and HCI effects. In *2010 NASA/ESA Conference on Adaptive Hardware and Systems*. IEEE, 215–222.

[28] Andrew Stern, Dhwani Mehta, Shahin Tajik, Farimah Farahmandi, and Mark Tehranipoor. 2020. SPARTA: A laser probing approach for trojan detection. In *2020 IEEE International Test Conference (ITC)*. IEEE, 1–10.

[29] Takeshi Sugawara, Daisuke Suzuki, Ryoichi Fujii, Shigeaki Tawa, Ryohei Hori, Mitsuru Shiozaki, and Takeshi Fujino. 2014. Reversing stealthy dopant-level circuits. In *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 112–126.

[30] Pawel Swierczynski, Marc Fyrbiak, Philipp Koppe, Amir Moradi, and Christof Paar. 2017. Interdiction in practice—Hardware Trojan against a high-security USB flash drive. *Journal of Cryptographic Engineering* 7, 3 (2017), 199–211.

[31] Shahin Tajik, Heiko Lohrke, Jean-Pierre Seifert, and Christian Boit. 2017. On the power of optical contactless probing: Attacking bitstream encryption of FPGAs. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 1661–1674.

[32] Nandeesha Veeranna and Benjamin Carrion Schafer. 2016. Hardware Trojan detection in behavioral intellectual properties (IP's) using property checking techniques. *IEEE Transactions on Emerging Topics in Computing* 5, 4 (2016), 576–585.

[33] Xinmu Wang, Seetharam Narasimhan, Aswin Krishna, Tatini Mal-Sarkar, and Swarup Bhunia. 2011. Sequential hardware trojan: Side-channel aware design and placement. In *2011 IEEE 29th International Conference on Computer Design (ICCD)*. IEEE, 297–300.

[34] Kaiyuan Yang, Matthew Hicks, Qing Dong, Todd Austin, and Dennis Sylvester. 2016. A2: Analog malicious hardware. In *2016 IEEE symposium on security and privacy (SP)*. IEEE, 18–37.

[35] Xuehui Zhang and Mohammad Tehranipoor. 2011. Case study: Detecting hardware Trojans in third-party digital IP cores. In *2011 IEEE International Symposium on Hardware-Oriented Security and Trust*. IEEE, 67–70.