Key Extraction Using Thermal Laser Stimulation

A Case Study on Xilinx Ultrascale FPGAs

Heiko Lohrke*[1], Shahin Tajik*†[3], Thilo Krachenfels[2], Christian Boit[1] and Jean-Pierre Seifert[2]

[1] Semiconductor Devices Group
[2] Security in Telecommunications Group
Technische Universität Berlin, Germany
lohrke@mailbox.tu-berlin.de, christian.boit@tu-berlin.de
{tkrachenfels,jpseifert}@sect.tu-berlin.de
[3] Florida Institute for Cybersecurity Research
University of Florida, USA
stajik@ufl.edu

Abstract. Thermal laser stimulation (TLS) is a failure analysis technique, which can be deployed by an adversary to localize and read out stored secrets in the SRAM of a chip. To this date, a few proof-of-concept experiments based on TLS or similar approaches have been reported in the literature, which do not reflect a real attack scenario. Therefore, it is still questionable whether this attack technique is applicable to modern ICs equipped with side-channel countermeasures. The primary aim of this work is to assess the feasibility of launching a TLS attack against a device with robust security features. To this end, we select a modern FPGA, and more specifically, its key memory, the so-called battery-backed SRAM (BBRAM), as a target. We demonstrate that an attacker is able to extract the stored 256-bit AES key used for the decryption of the FPGA's bitstream, by conducting just a single non-invasive measurement. Moreover, it becomes evident that conventional countermeasures are incapable of preventing our attack since the FPGA is turned off during key recovery. Based on our time measurements, the required effort to develop the attack is shown to be less than 7 hours. To avert this powerful attack, we propose a low-cost and CMOS compatible countermeasure circuit, which is capable of protecting the BBRAM from TLS attempts even when the FPGA is powered off. Using a proof-of-concept prototype of our countermeasure, we demonstrate its effectiveness against TLS key extraction attempts.

Keywords: Bitstream Encryption · FPGA Security · Thermal Laser Stimulation

1 Introduction

The primary motivation behind reverse-engineering of embedded devices is to get access to the utilized intellectual property (IP) to counterfeit and overbuild the target products [VT13, TGB17] or to extract other sensitive information contained within. One of the main approaches to tamper with a design or clone its contents is carrying out physical attacks, such as side-channel analysis. One class of side-channel analysis approaches is referred to as optical attacks, which are mostly relying on known failure analysis (FA) techniques. In [NSHB13], an optical attack using thermal laser stimulation (TLS) was

*These authors contributed equally to this work.
†During the course of this study, Shahin Tajik was with the Security in Telecommunications Group of the Technische Universität Berlin, Germany.

IACR Transactions on Cryptographic Hardware and Embedded Systems ISSN 2569-2925, Vol. 2018, No. 3, pp. 573–595
DOI:10.13154/tches.v2018.i3.573-595

introduced. This attack demonstrated the potential of TLS to localize the key storage (e.g., SRAM) of chips and extract its contents.

In contrast to many other side-channel techniques, TLS potentially enables the adversary to obtain secret information from the chip by conducting just a single measurement. However, previous work only showed detectability of data changes in the TLS measurement and not actual data extraction [NSHB13, BHNF13]. Moreover, the demonstrated changes in the TLS pattern were not discussed beyond single bits. The effectiveness of this optical attack has furthermore been evaluated against microcontrollers, which in comparison to recent IC technologies were built in relatively large technologies [NSHB13, BHNF13]. Hence, the question is raised whether it is feasible to actually extract secret data using TLS, especially on more advanced and complex ICs manufactured in smaller technologies. Besides, it is unclear, whether the same attack can be applied against a device provided with state-of-the-art protection schemes.

The main aim of this work is therefore to evaluate the practicability of extracting sensitive information by an adversary, who seeks to attack a modern platform with advanced security schemes and has no knowledge about the underlying hardware design. To this end, a suitable target platform with strong security features has to be selected. Recent generations of field programmable gate arrays (FPGAs) seem to be appropriate candidates for this, as FPGA vendors have been aware of physical attacks and have integrated several countermeasures into their products to protect the deployed IP. Moreover, FPGAs have become vital parts of several consumer, industrial and military applications. We chose a device from Xilinx's Ultrascale Series FPGAs manufactured with 20 nm technology to present our attack [Wil17]. Ultrascale FPGAs are considered to be secure, and to the best of our knowledge, their IP protection schemes have not been broken yet. This is due to the implemented side-channel countermeasures on these devices, which make a wide range of physical attacks ineffective.

Our Contribution. In this work, we present how an attacker can deploy TLS to localize the battery-backed RAM (BBRAM) of an FPGA and read out the stored key inside it in just a *single measurement* and in a *non-invasive* way from the IC backside. During the proposed attack the FPGA is powered off, and thus, implemented countermeasures during configuration or runtime of the chip are ineffective. Based on our time measurements, an attacker needs less than 7 hours to localize and reverse-engineer the key storage. We additionally show that key extraction can be performed either visually or automatically and prove the reliability of automated extraction. We further discuss the feasibility of the TLS attack on different platforms in different scenarios. Afterward, we propose a countermeasure to remedy the vulnerability of SRAM cells against TLS attacks. Finally, we evaluate the effectiveness of our countermeasure approach by using a proof-of-concept circuit in conjunction with the Ultrascale FPGA device. It should be noted that conducting TLS from the IC backside has been previously reported in the literature and is not the contribution of this work. Our primary intention was to draw attention to the potential threat of this known but not well-researched technique by presenting a case study.

Xilinx has been informed about this security weakness in December 2017. It should be noted that the presented vulnerability in this work is not limited to Xilinx devices. Not only FPGAs from Intel (formerly Altera) and Microsemi, which utilize BBRAM or SRAM physically unclonable functions (PUFs), but all ICs employing SRAMs are in principle unguarded against this kind of attack.

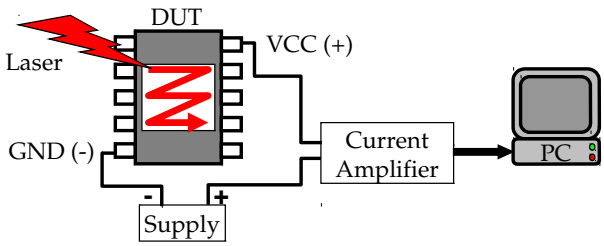


Figure 1: Supply current monitoring laser stimulation setup. Depending on the laser wavelength, thermal or photoelectric stimulation can be applied. Figure based on [BDPL01].

2 Background

2.1 Laser Stimulation Techniques

In failure analysis (FA) a number of techniques have been developed, which use laser radiation to influence the device under test (DUT). These techniques are referred to as laser stimulation techniques and measure changes in device parameters in response to the incident laser radiation. This is usually performed by scanning a region of interest with a laser beam and monitoring the device parameters, see Fig. 1. As the stimulated area of the DUT behaves differently when it is affected by faults, this allows FA engineers to isolate the fault cause. The stimulation effect employed depends on the laser wavelength. If the photon energy is larger than the silicon bandgap, photocarriers will be generated in the semiconductor. Otherwise, the effect of the stimulation is mainly localized heating. While the former technique is referred to as photoelectric laser stimulation (PLS), the latter is called thermal laser stimulation (TLS). Both the photocarriers as well as the heating then influence the parameters of the device. The monitored parameters are often simply voltage or current at a specific device pin.

For instance, for a current monitoring laser stimulation (see Fig. 1), the device is biased with a specific supply voltage, and the current between the supply pins is monitored via a current preamplifier. The exposed silicon backside is then scanned by a laser beam which penetrates the silicon and stimulates the structures inside the device. A PC simultaneously samples the current preamplifier output and plots it as a 2D map of the device's response to the stimulation. An example of a current monitoring TLS technique is optical beam induced resistance change (OBIRCH), which can be used to localize shorts in a device. To conduct this experiment, the device is biased with a small voltage, and the current flowing is monitored. When the thermal stimulation heats up a part of the device, its resistance will change due to the temperature change. If that part is carrying a significant portion of the short-circuit current, this will change the overall current and can be detected. As the current change is often most pronounced at the cause of the short, it can easily be localized.

Although TLS and PLS have mainly been employed to localize defects in the past, they have been used as attack techniques as well. It has been demonstrated that using PLS an adversary can extract secret information from a chip. The presented attacks in [SSAQ02, Sko06] have been mounted against common microcontrollers built in large technologies (e.g., 900 nm). Moreover, the reported attacks have been carried out from the IC frontside, which makes it impractical against modern chips due to the existence of multiple metal layers on the IC frontside. In another work, a proof-of-concept attack using TLS has been presented. However, the authors did not interpret the resulting patterns apart from single bits and did not demonstrate actual data recovery [NSHB13].

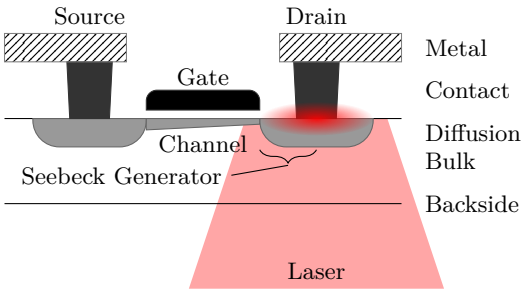


Figure 2: Seebeck voltage generation in a MOSFET transistor. Figure based on [BHNF13].

2.2 Thermal Laser Stimulation of SRAM

This section aims at giving the necessary background information on SRAM thermal laser stimulation. To illustrate the data dependency of TLS data from SRAM memory cells, it is first necessary to understand the behavior of a single MOSFET transistor under stimulation, which is shown in Fig. 2. In this case, the drain of the transistor is heated by the laser beam, creating a temperature gradient, which in turn causes a diffusion of carriers. As discussed in detail in [BHNF13, NSHB13], this effectively leads to the generation of a voltage source between the drain's metal contact and the channel of the MOSFET, which is also referred to as a "Seebeck generator". If the transistor is on (low-ohmic channel), this Seebeck generator is basically connected between source and drain. If it is off (high-ohmic channel) one terminal of the Seebeck generator is effectively floating. A corresponding situation occurs if the source is stimulated. However, the voltage generated between source and drain will have an opposite sign, due to the changed orientation of the temperature gradient [NSHB13]. It can thus be seen that the Seebeck generator can only act on devices connected to the source or drain if there is a low-ohmic channel. Furthermore, the sign of the voltage will change depending on which side of the MOSFET is stimulated.

With these principles in mind, it is now possible to analyze the data dependency of the TLS response of a single SRAM cell. Fig. 3 shows a schematic of the basic memory element of a typical SRAM cell consisting of two cross-coupled inverters. For simplicity, the connections and transistors for read/write access have been omitted. It can be seen that this circuit will keep one of two states due to the cross-coupling. If, as shown in the schematic, the lower left NMOS transistor is on, it will pull the gates of the right-hand side transistors low. This will turn the top right PMOS transistor on and the lower right NMOS transistor off, which keeps the gates of the left-hand side transistors at a high level, thus keeping the cell's state stable. It is evident that by applying suitable voltages to the gates of the transistors, the cell can be flipped into its inverted state. In this case, the top left PMOS and the lower right NMOS would be on, and the cell's state would be kept stable by the same mechanism. These two states can then be used to store either a "1" or a "0" bit in the SRAM cell. It is evident that for ideal transistors there would be no current draw between VCC and GND when the cell is stable.

However, if thermal stimulation is applied to the cell shown in Fig. 3, Seebeck voltages ($U_{Seebeck}$) will be generated at the transistors. Following the previous reasoning, for high-ohmic transistors the Seebeck generators will have no chance to act upon the circuit, as they are floating. However, the voltages generated at the top right PMOS and the bottom left NMOS can. For example, a stimulation at the highlighted drain of the top right PMOS will cause the voltage at its drain to decrease to $VCC - U_{Seebeck}$. This will then act upon the gate of the top left PMOS (red arrow), causing a slight decrease in the resistance of its channel via exponential sub-threshold operation. As the lower left NMOS is already on, this will lead to a leakage current between VCC and GND. This increase in

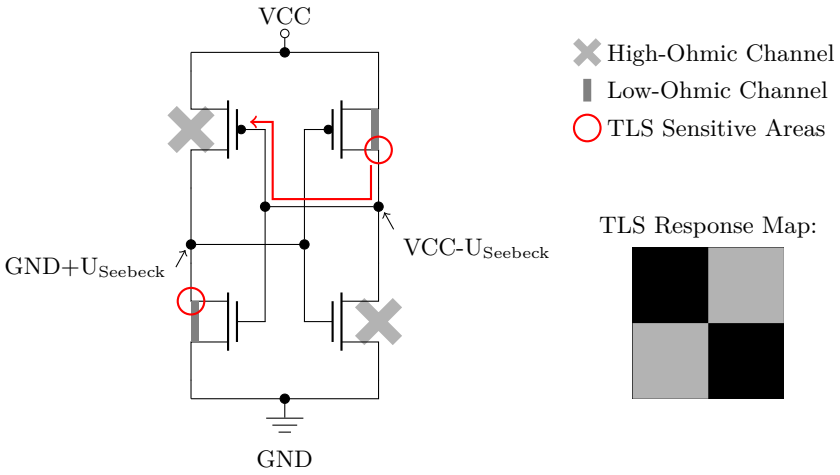


Figure 3: SRAM cell under thermal stimulation and expected simplified TLS response map under the assumption of transistor size ≈ beam diameter. The TLS response map will be inverted when the cell's bit is toggled. Figure based on [BHNF13].

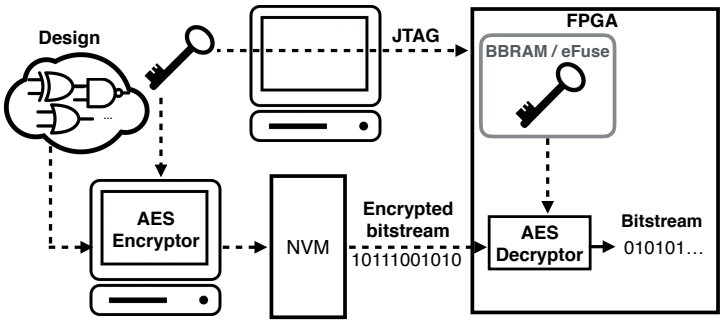


Figure 4: Encrypting the bitstream in the IDE using a key and decrypting it on the FPGA by an ASIC decryption core using the same key.

current can then be detected at the supply pins. Because the cell is symmetric, a similar behavior can be observed when stimulating the drain of the bottom left NMOS. In this case, the decreased resistance of the lower right PMOS would also cause an increased VCC-GND leakage current. As a result, it can be reasoned that if the area of the SRAM cell is scanned in a TLS setup and the current consumption is plotted over the X/Y position, a TLS response map similar to the one depicted in Fig. 3 can be expected. If the laser beam diameter is approximately equal to the transistor size, the areas of the sensitive transistors will be brighter due to the grayscale-encoded current consumption. The insensitive transistors, on the other hand, will be darker. From this TLS map, the current bit state of the cell can be deduced, as the opposite bit state would have an inverted TLS response map.

2.3 FPGA Security during Configuration

Since SRAM-based FPGAs do not contain any non-volatile memory (NVM) to store the bitstream [TM14] (i.e., the configuration data), the bitstream has to be kept in an external NVM and loaded into the FPGA upon each power-on in an adversarial field. In order to keep the bitstream confidential and prevent IP piracy, modern FPGAs are deploying

bitstream encryption schemes. In this case, a secret key is used to encrypt the bitstream in a trusted environment by the integrated development environment (IDE) software. Afterward, the bitstream is stored in the external flash memory on the same board, see Fig. 4. At the same time, the secret key is transferred to the FPGA and stored either in the battery-backed RAM (BBRAM) or the eFuses inside the FPGA. By powering the FPGA in the field, the encrypted bitstream is transmitted to the FPGA and decrypted inside the device by a dedicated ASIC decryption core using the already stored secret key, see Fig. 4. Recent generations of FPGAs from different vendors deploy AES-256 to encrypt the bitstream, although the mode of operation might differ between devices. To reverse-engineer or clone a running application on an FPGA, the attacker needs to have access to the unencrypted bitstream. Therefore, the attacker needs to extract either the key or the plaintext bitstream from the chip.

2.3.1 Key Storage Security

The secret key for the decryption of the bitstream can be stored in the BBRAM or eFuses inside the FPGA [Wil17, LKA18]. There are no readback paths for the BBRAMs and eFuses, and thus, the stored keys in these memories cannot be read out. BBRAM is considered more secure than eFuses [Wil17, LKA18] since the BBRAM can be zeroized if any tampering attempt is detected during runtime. Moreover, in contrast to the eFuses, the stored key in the BBRAM can be updated during the lifetime of the product several times. However, BBRAMs are requiring an external battery to maintain the key data when the device is powered down. These batteries must be guaranteed to be operational for several years, and therefore, the associated reliability issues are a downside of this kind of storage.

Since both memory technologies are non-volatile, they can in principle be the target of an attack when the chip is powered off. It is reported that the stored key in eFuses can be read out by scanning electron microscopy (SEM) [TM14]. However, the BBRAMs of FPGAs have not been the target of attacks in the literature. Hence, it is still unclear whether they can be considered secure or not.

2.3.2 Decryption Core Security

It has been reported that the decryption cores of several families of FPGAs are vulnerable to differential power analysis (DPA) [MS16, SMOP15]. By mounting DPA against the decryptor, the secret key can be extracted and used to decrypt the encrypted bitstream stored in the NVM. Moreover, it has been shown that by conducting optical probing, the attacker is able to probe the bitstream after the decryption on the chip [TLSB17]. To mitigate the shortcomings of the decryption core against such physical attacks, FPGA vendors have integrated different countermeasures to protect the key and plaintext bitstream.

Since DPA and optical probing attacks require several repeated measurements to acquire an adequate signal to noise ratio, they become ineffective by implementing a so-called DPA counter, which limits the number of reboot attempts on an FPGA [Wil17]. If the number of rebooting attempts exceeds the predefined threshold, the key in the BBRAM can be zeroized to prevent further leakage of information. The DPA counter has been integrated into Xilinx Ultrascale/Ultrascale+ devices. Moreover, authentication of the encrypted bitstream can defeat DPA with randomized data. To avert DPA with authentic data, key rolling can be deployed, where the secret key is updated several times during configuration. Both authentication and key rolling techniques have been implemented on Xilinx Ultrascale/Ultrascale+ and Intel Stratix 10 devices [Wil17, LKA18].

3 Attack Scenario

For our attack scenario, we assume that an attacker is in possession of a board containing an FPGA in a flip-chip package which uses an encrypted bitstream. We assume that the FPGA is additionally secured with RSA authentication and DPA countermeasures, such as boot/configuration counters. As this makes DPA [MS16] and optical contactless probing [TLSB17] infeasible, the attacker aims to extract the decryption key by using the TLS technique outlined in Sect. 2.2 to be able to either clone, modify, or reverse engineer the bitstream. Furthermore, she might also be interested in extracting secrets (e.g., authentication keys) from the encrypted configuration data. We assume that she has access to a laser scanning microscope (LSM) equipped with a laser suitable for TLS. This infrared LSM will most likely be rented by her at a failure analysis lab or similar facility such as a university. The LSM either is already equipped with a current preamplifier for TLS measurements, or a suitable device is acquired by herself and brought to the lab.

The BBRAM memory constitutes an ideal scenario for an attacker using TLS for key extraction. As the BBRAM usually is the only structure connected to the battery power supply, a very low amount of noise can be expected. Furthermore, the relatively low number of key bits makes it potentially feasible (though tedious) to extract the key manually. A further advantage is the ability to perform the key extraction with the board completely powered down, except for the battery supply. As no other components will be active, the noise will be further lowered and the FPGA will additionally be unable to take defensive actions, if present. A challenge will, however, be the small memory cell size expected for recent technology devices. To extract the key from the device, the attacker will have to execute the following steps: (1) locate the BBRAM key memory inside the device, (2) verify that the BBRAM stimulation response is key dependent, (3) determine the physical location of all key bits in the BBRAM, (4) extract the key, either visually from the response map data or automatically by developing image recognition tools. To be on the safe side, she will probably perform these steps on a training device of the same type, before moving to her actual target.

To locate the BBRAM, she will first acquire optical images from inside the device to gain an overview of the die layout. Because of the flip-chip package, which is transparent to the infrared wavelength employed, this can be achieved without any package or board modification. Using datasheets and similar technical information, such as floorplans available in the FPGA design software, she can then determine candidates for the configuration logic area position. This area will most likely contain the decryption logic and the BBRAM key storage with it. If she is not able to find suitable candidate locations, she will simply have to examine the whole device in the following step.

After having determined the search area, she will apply laser stimulation to it. First, she connects the BBRAM supply pins to her current preamplifier and lets the laser scan over the area of interest, compare Fig. 1. As soon as the laser hits the actual BBRAM area, it will influence the current at the supply pins. This area of changed supply current will then be apparent in the 2D response map displayed on the PC of her setup. As soon as she has found the BBRAM area in this way, she can examine it for data dependency of the stimulation response map. For this, she will program different keys into the memory of her training device and acquire TLS responses. In this response map, she should be able to see changes when she changes the data value of individual BBRAM cells, compare Fig. 3. If she sees these pattern changes caused by data changes, it means that she will also be able to deduct the data contents from the pattern.

For this, however, she has to determine the location and possibly the orientation of the individual key bits. This process, referred to here as logical-to-spatial mapping, can be carried out in a straightforward way. As she can set individual bits in the key of her training device, she can examine where they lead to changes in the response map. To aid her in this process she might also use difference calculation between an all-zeroes or

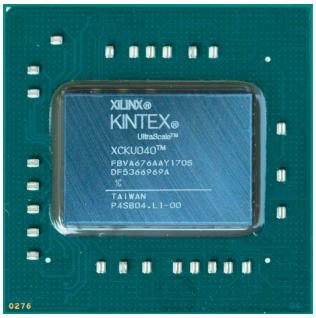


Figure 5: Image of a Xilinx Ultrascale XCKU040-1FBVA676 device in a flip-chip BGA package [AVN18]. The exposed silicon backside of the die can be seen in the middle of the package.

all-ones key reference and her current measurement. This will highlight only the changed locations and allow her to identify the current bit's physical position more easily. In the worst case, this will require the same amount of TLS measurements as there are key bits. Yet, the mapping of the key bits might follow logical rules, and she might thus be able to determine it with fewer measurements.

When she has determined the mapping, she can move on to extract the data. One option is to do this manually. For this, she can directly use her knowledge about the pattern shape and the key bit mapping to analyze a TLS measurement visually. Alternatively, she might also subtract a reference measurement (e.g., all-zeroes key) to see the set bits more clearly. The second option is to develop image recognition software to automatically recover the key from TLS data. This could be advantageous for her if manual extraction is too tedious, for example, if she wants to extract keys from multiple devices or the key has too many bits. To accomplish this task, she can employ her knowledge about the cell positions from the logical-to-spatial mapping and write software to analyze the pattern in each cell.

When she has established a working key extraction process this way, she will finally move to her target device. In this case, she will connect the target device to her setup and then extract its key using either the manual or automatic method. To avoid key loss during this critical part, she will solder an "emergency battery" to the BBRAM supply pins before removing the original battery. This emergency battery is merely a standard backup battery connected in series with a diode in conducting direction. In this way, as soon as the original battery is removed from its holder, the BBRAM will be supplied from the emergency battery. The current amplifier is then connected in parallel and set to a higher bias voltage than the emergency battery voltage. In this way, during the experiments, all current will be sourced from the preamplifier and no current can flow from the amplifier into the emergency battery. If there are any connection issues or other problems during the experiments and the voltage drops, the emergency battery will start to supply the BBRAM through the diode. This will make key loss during the final extraction very unlikely.

4 Experimental Setup

4.1 Device Under Test

We chose a Xilinx Ultrascale FPGA development board designed by AVNET as the target platform (model AES-KU040-DB-G). It contains a Xilinx Ultrascale XCKU040-1FBVA676 FPGA manufactured with 20 nm technology in a flip-chip ball grid array (BGA) package, see Fig. 5. The silicon die is inverted and placed frontside down in this type of package.

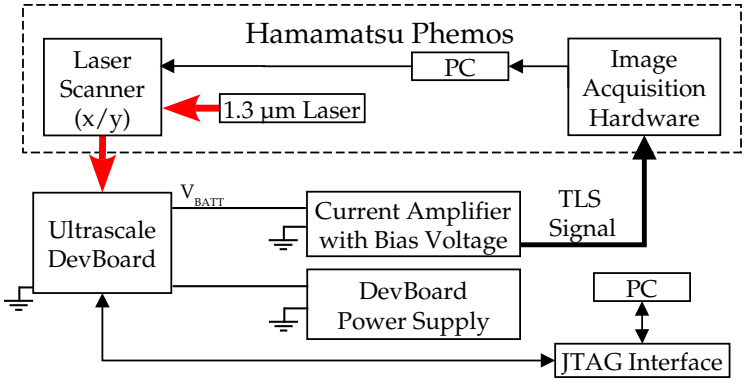


Figure 6: Block diagram of the setup used for laser stimulation.

Hence, we have direct access to the silicon substrate on the backside of the chip. Based on our measurements, the thickness of the substrate is about 750 μm. An image of the die can be acquired without any substrate thinning by selecting a light source with a wavelength to which the silicon is transparent (1.3 μm), see Fig. 7. Hence, to conduct a laser stimulation attack from the backside of the chip, no preparation is required.

The device contains an advanced encryption standard (AES) decryption core ASIC with a 256-bit decryption key. Xilinx Ultrascale Series FPGAs are using AES in Galois/Counter Mode (GCM) mode to encrypt the bitstream [Wil17]. As discussed in Sect. 2.3.1, the key can be stored in battery-backup RAM (BBRAM). A battery is used to keep the key in memory when the main power supply is removed [Wil17]. The bitstream data can either be placed in flash memory and transferred via SPI to the FPGA or can be directly loaded via a JTAG interface. The JTAG interface can also be used to configure the internal settings of the device.

4.2 Electrical and Optical Setup

Fig. 6 shows the setup used for stimulation response map acquisition. The setup consists mainly of a Hamamatsu "Phemos-1000" laser scanning microscope and a Stanford Research Systems "SR570" current amplifier. A 1.3 μm laser which delivers a maximum power of 21 mW (5x lens) or 13 mW (50x lens) onto the DUT is part of the Phemos system (model C10656).

The Ultrascale development board is placed inside the Phemos, and a PC is connected to the JTAG interface to allow access to the BBRAM, configuration memory, and general device settings. When actively being used, the board is mainly powered by the provided development board supply, which then also powers the BBRAM. However, in the case of the board being powered down, the BBRAM is supplied via the backup battery connection V_{BATT}. This V_{BATT} voltage is supplied by the current amplifier's bias feature in our setup. The amplifier is connected via a coaxial cable soldered to the battery holder. No other electrical modifications were performed on the board. The current amplifier converts the V_{BATT} current flow into a proportional voltage, which constitutes the actual TLS signal during stimulation. This signal is fed into an auxiliary input of the Phemos image acquisition hardware, where it is analog-to-digital converted. The SR570 bias voltage can be set to nominal battery levels, allowing for normal BBRAM operation while monitoring the current.

During stimulation response acquisition, the laser is focused through the silicon backside of the DUT to reach the active area. For this, either a 5x/0.14 NA or a 50x/0.71 NA lens is used. The 50x lens is equipped with a correction feature for silicon substrate thickness.

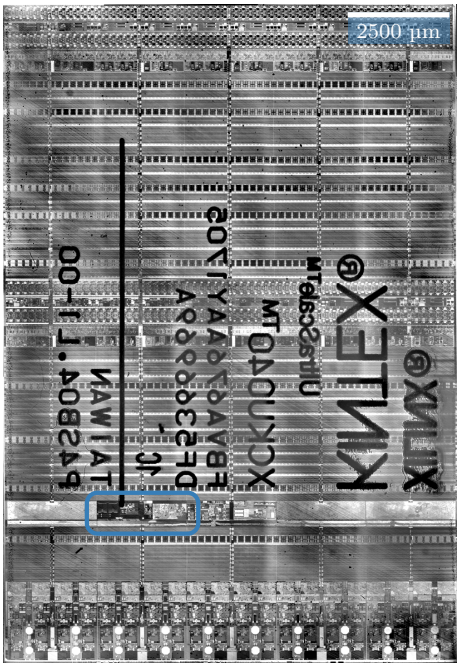


Figure 7: Overview reflected light image of the Xilinx Ultrascale XCKU040 die. The area containing the configuration and decryption logic is highlighted.

During stimulation, the FPGA is powered down except for the battery backup voltage. The beam is then moved over a region of interest using galvanometric scan mirrors while the output signal of the current preamplifier is sampled simultaneously. This data can then be assembled into a 2D representation of the BBRAM current consumption in accordance with the laser beam position. This visualization then constitutes the stimulation response map of the device.

5 Results

5.1 BBRAM Localization

For finding the BBRAM, the approach discussed in Sect. 3 is used. First, optical overview images of the device are acquired. Fig. 7 shows a die overview image assembled from multiple reflected light images. The assembly was performed using stitching software [PST09]. The area of stimulation can be limited to the configuration and decryption logic area whose position can be deduced from information given in datasheets and similar documentation. In our case, we compared the floorplan information, available in the "Device" window of the "Vivado" IDE [Xil12], to the optical images. The area that was determined to be the configuration logic is highlighted in Fig 7.

To prepare for TLS measurements in this area, a key was loaded to the BBRAM over JTAG and the board was powered down. The BBRAM battery voltage V_{BATT} was supplied by the current amplifier set at 1.5 V bias. Measurements were then performed using the 5x lens, 40% laser power, 20 nA/V current amplification, and 72 s scan time. Fig. 8 shows a thresholded overlay of the resulting stimulation response onto a reflected light image. It can be seen that multiple locations in the center show a response to stimulation.

This area was thus further examined using the 50x lens. Fig. 9a shows the resulting TLS map. It can be seen that two structures are responsive to the stimulation, one on the left

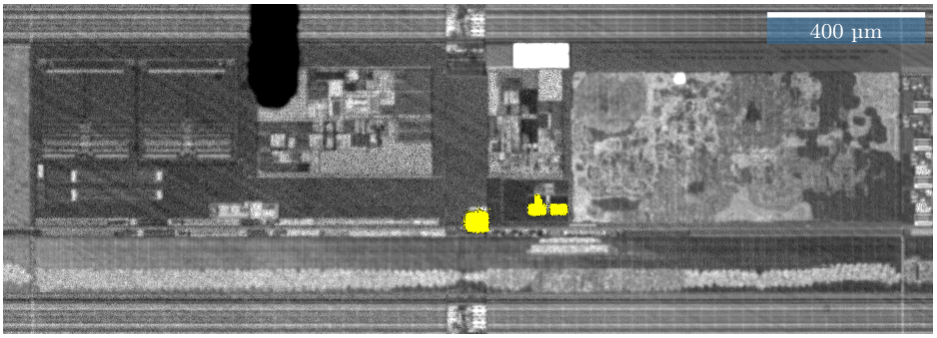
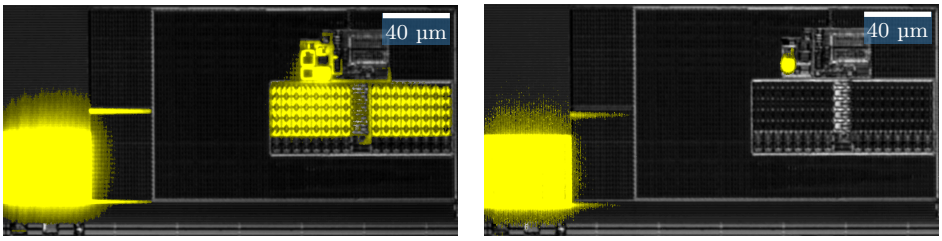


Figure 8: TLS measurements for key memory localization using the 5x lens.



(a) BBRAM key storage active.

(b) BBRAM key storage inactive.

Figure 9: TLS measurements for key memory localization using the 50x lens.

and one on the right-hand side. Closer examination revealed that only one of the structures showed dependency on deactivation of BBRAM key storage, compare Fig. 9b. This area was thus assumed to contain the BBRAM and examined further. The other structure was disregarded and considered to be an electrostatic discharge protection structure for the battery input. Fig. 10 shows a detailed reflected light image of the suspected BBRAM area. Its structure is similar to the one to be expected for standard SRAM structures with word line and bit line access structures in the middle and bottom and identical cells seemingly distributed in two blocks.

5.2 Data Dependency of Response Map

TLS measurements performed in the lower half of Fig. 10 revealed a dependency of the stimulation response on key data. This confirmed that this area indeed contains the

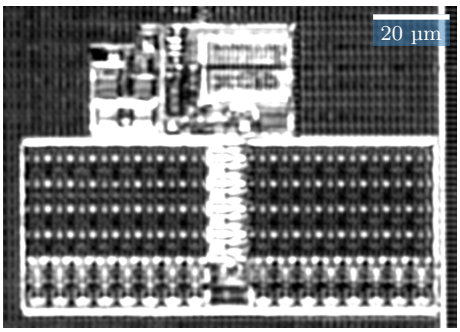


Figure 10: Reflected light image of the BBRAM AES key storage.

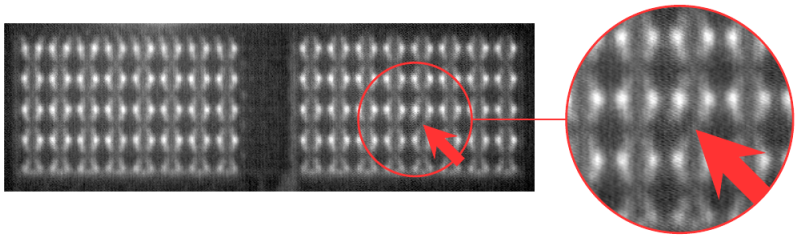


Figure 11: Data dependency of the TLS response. A single bit (bit 120) has been set in the BBRAM key data which manifests as an irregularity in the measurement result.

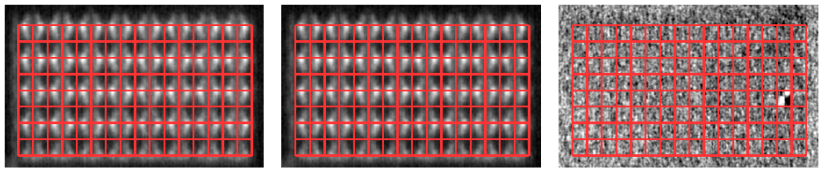


Figure 12: Difference calculation between an "all bits zero" TLS reference and measurement data quickly reveals which bits are set in the AES key. As an example the right-hand half of the BBRAM with a single bit set (bit 126) is shown here.

BBRAM.

Fig. 11 shows a TLS response map of this area with a single bit set in the key data. It can be seen that the set bit manifests as an irregularity in the TLS data, see Fig. 11. Judging from the surrounding cells, the expected TLS signal for a zero bit in this cell would be a line from top left to bottom right. Instead, a line from bottom left to top right is seen. Apart from the bright dot at the top, this is in accordance with the expected simplified TLS response illustrated in the background section (see Fig. 3). To make this change even more evident, a reference measurement with an "all zeroes" key can be subtracted from the TLS data for the current key. Fig. 12 shows an example of this procedure performed on only the right-hand half of the BBRAM, with an overlay of the assumed key memory cell boundaries. The single bit that is set can clearly be identified in the difference image. Data dependency of the TLS response map on the key bits is thus proven.

5.3 Logical-to-Spatial Mapping

Analysis of the memory area to obtain the corresponding location of each key bit was then carried out systematically. This process, referred to as "logical-to-spatial mapping", revealed that the mapping is straightforward. The key bit index simply increases from left to right and from bottom to top. However, it should be noted, that the orientation of the cells is flipped for every row and column, compare Fig. 11 and 12. It also became apparent that there is an additional row of memory (32 bits) at the top of the BBRAM, compare Fig. 12. It seems that this area is used for security-relevant features. This assumption is due to the fact that its data changes when enabling different features such as the so-called "DPA counter" of the device. Comparison with source code made publicly available by Xilinx as "XilSKey Library" [Xil18c] for key programming via JTAG, revealed that these 32 bit are most likely the so-called "control word". This control word is used for storing settings of features such as the DPA countermeasures and obfuscated key storage. Information from the source code was then combined with assuming the same straightforward mapping as for the key bits to arrive at a hypothesis for the control word mapping. This mapping was then confirmed using further TLS measurements and changes

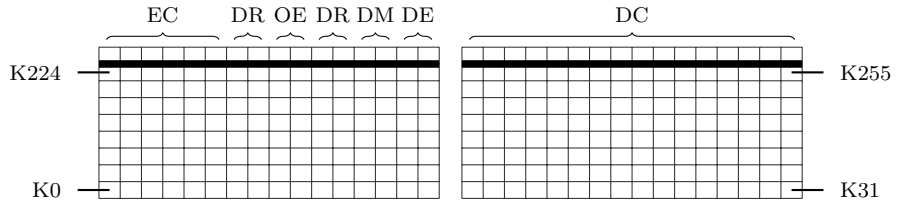
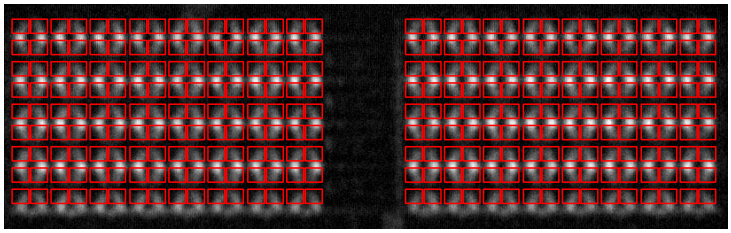
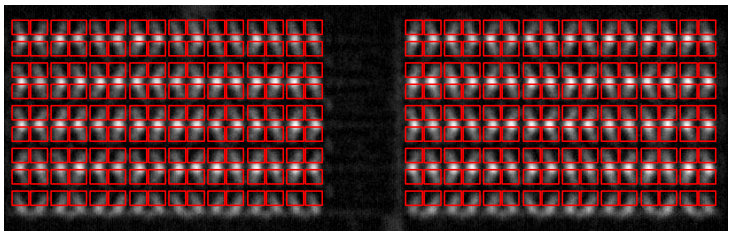


Figure 13: Mapping of the individual AES key bits (K) in the BBRAM. The bit number increases from left to right and bottom to top. Additionally, other security-relevant data is saved at the top row: EC = error check, DR = DPA countermeasure reserved bits, OE = key obfuscation enable, DM = DPA countermeasure mode, DE = DPA countermeasure enable, DC = DPA countermeasure counter (8 bit, redundant).



(a) Reference image of the cleared BBRAM



(b) Target image containing the key

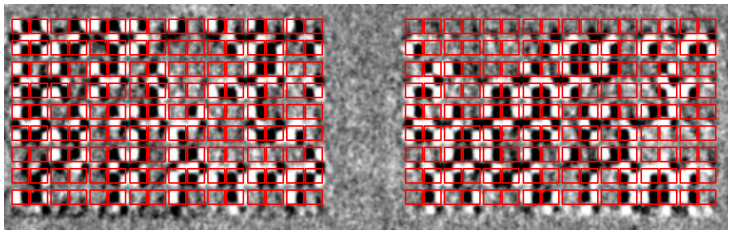
Figure 14: Reference and target image

to the respective device settings via JTAG. Fig. 13 gives an overview of the determined BBRAM mapping for the key as well as for the control word. With this information, visual key recovery from a TLS response map would be possible for an attacker.

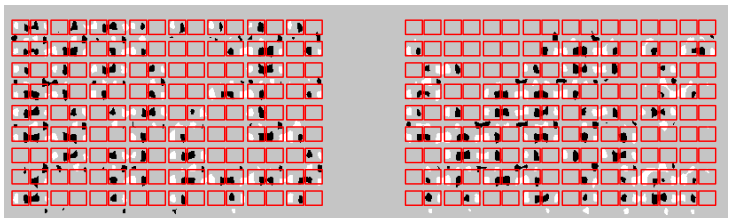
5.4 Automatic Key Recovery

To evaluate automatic data extraction, we implemented a script which is able to determine all bits stored in the BBRAM. Following the idea explained in Sect. 3 and already applied in Fig. 12, a difference image between the zeroized BBRAM (reference image, Fig. 14a) and an image containing the actual key (target image, Fig. 14b) is calculated. Before creating the difference image (Fig. 15a), the target image is automatically aligned to the reference image and the contrast is enhanced. After applying a Gaussian blur filter and removing the background using a rolling ball filter, the difference image is thresholded twice (Fig. 15b): once in the positive and once in the negative direction to utilize both the bright and the dark spots of the difference image. The described steps are all performed using plugins available in the *Fiji* [SACF+12] image manipulation program by means of a macro script. For alignment, the Linear Stack Alignment plugin is used [Low04]. For thresholding, Otsu's method is applied [Ots79].

A Python command line tool (≈ 300 lines, PEP-8 formatted) first calls the macro



(a) Difference of reference and target image



(b) Combined threshold image of bright and dark spots in the difference image

Figure 15: Difference images

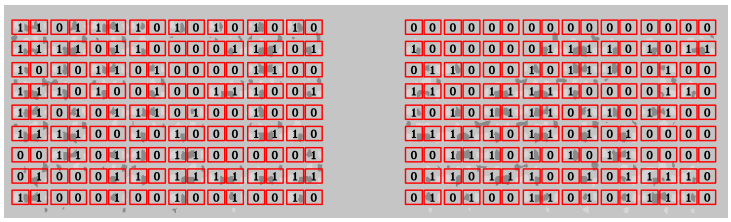


Figure 16: Decoded bit values of a random key. Control bits: `0x0000557b`, Key: `0xd781b86f274630b561f39c9736f512eb0adf714f0d5c836c7a76ff627aca4923`

script (≈ 80 lines) and then analyzes the resulting data. For analysis, the OpenCV library [Its18] is used. Decoding is done iteratively over a grid, where each cell represents that part of the threshold images which contains the information for exactly one bit. The bit values (Fig. 16) then can be extracted as follows: the value is `1` if both threshold images contain a certain amount of non-zero pixels, otherwise, the bit value is `0`.

Finally, a spatial-to-logical conversion maps this data to all key and control bits. To evaluate reliability, we performed TLS measurements using five random keys, an all-zero, and an all-one key. For these measurements, 40% laser power, 20 nA/V amplification, and 72 s scan time were used. Our script was able to extract all key and control bits correctly in all cases.

5.5 Time Expenditure

To measure the required time that the attacker might need to rent the necessary equipment we have used time tracking software on the PC of the laser stimulation setup. This revealed that the experiments needed for BBRAM localization and key bit mapping took a total of about 7 hours. Using the knowledge gained and the image recognition software presented, a key can be now be extracted from the device in less than 15 minutes.

6 Discussion

6.1 Obfuscated Key Storage

Xilinx's Ultrascale devices enable the user to load the secret key into the FPGA in an obfuscated format [Wil17]. In this case, the obfuscated key is generated in the IDE by encrypting the red key (i.e., unencrypted key) with a metalized family key, which is stored permanently in the FPGA. There is one metalized key for all Ultrascale devices. When loading the obfuscated key into the BBRAM, a flag is set in the BBRAM (see Fig. 13) indicating that this key is obfuscated, and thus, it should be decrypted before the decryption of the bitstream is started. Similarly, some Intel Series 10 FPGAs employ proprietary techniques to obfuscate the key and store the wrapped key in the BBRAM [Int17, LKA18]. Obfuscation enables the user to give the obfuscated key to a contract manufacturer without exposing the red key [Wil17].

In this case, if the attacker reads out the key stored in the BBRAM using TLS, she will have only access to the obfuscated key. Since she does not possess the metalized key, she cannot unwrap the obfuscated key, and consequently, she is not able to decrypt the stored bitstream in the NVM. However, although obfuscation prevents the attacker from reverse-engineering the design or tampering with it, it is still possible for her to clone the design without knowing its content. In other words, the attacker can copy the encrypted bitstream to the NVM of another FPGA board, and then load the obfuscated key as well as the "obfuscation enable" flag into its BBRAM. Since there is one metalized key for the whole device family, the obfuscated key is unwrapped correctly, and therefore, the bitstream is decrypted as well and will configure the FPGA. It needs to be noted that activating RSA authentication also does not prevent cloning, since the public key, which is used to authenticate the bitstream, is stored in the plaintext part of the bitstream. As a result, the attacker can copy the same public key and program it to any other FPGA [Wil17].

One potential countermeasure against cloning when an obfuscated key is used could be including a check of the unique device identifiers (IDs) available in Ultrascale FPGAs. These IDs can be the device serial number ("device DNA" eFuse) or user-programmed identifiers ("user" eFuse) [Pet17]. For this, a check of the expected ID is included in the encrypted design bitstream. After configuration, before enabling the actual design functionality, the check is performed and a mismatch leads to deactivation. Depending on whether device DNA or user eFuses are used, the design can be locked to one or multiple devices [Pet17].

If obfuscated key storage is enabled, and assuming that the metalized key is unknown, the attacker will be unable to modify the bitstream to skip the check. However, if the check relies on user eFuses, she can first extract the obfuscated key from the BBRAM of the original device and then clear it. Additionally, she will obtain a copy of the original bitstream from NVM. This will allow her to load her own bitstream to NVM which then simply reads out the user eFuse data so she can copy it to the new device. The only way to prevent this is if the designer has enabled RSA authentication in "force" mode on the original device, as it prevents loading of unsigned bitstreams [Xil18b]. However, this feature increases configuration time by a factor of up to three, is incompatible with partial reconfiguration and serial, JTAG as well as most SPI configuration modes on Ultrascale devices. Additionally, the "force RSA authentication" mode prevents RMAs to be accepted by Xilinx [Xil18b].

If the designer has used the device DNA instead, a readout approach will not be fruitful, as it can not be altered in the new device. However, this would require the designer to generate a new bitstream for every single device, which might be tedious. Apart from this, the security of this approach depends heavily on the secrecy of the Ultrascale family metalized key.

6.2 Packaging

Our experiments were carried out on a bare-die flip-chip package, which advantageously requires no preparation at all for silicon access. Thus, the question of the applicability on other package types can be raised. Overmolded flip chips would simply require the removal of packaging epoxy while the chip is still on the board. Wire-bond devices can be removed, flipped, and milled. However, a challenge with them is that continuous power must be supplied during preparation. For pin-based packages, wires can be attached to the supply pins. With non-flipped ball-grid-array packages, the required ball might be under the device, making the attack much more challenging. Nevertheless, it should be stressed that the use of flip chips is on the rise for newer FPGA series due to performance issues.

6.3 TLS on other Platforms

Although we presented our TLS attack against the BBRAM of an FPGA, the same technique can be applied to SRAMs of any device. The main advantage of targeting BBRAM for an adversary is the low amount of noise on the supply rail of this memory. However, the attacker can still target SRAM cells which are supplied by the core voltage of the chip, and therefore, are only active when the chip is turned on. In this case, the attacker needs to measure the data-dependent leakage current on the core supply rail of the device. Since many other active circuits are supplied from the same voltage source, the noise level would be higher. Hence, the adversary might need to increase the number of her measurements and take an average of them to improve the obtained SNR. Besides, she can also reduce the scanning speed of the laser to further decrease the effect of noise.

Successful TLS data extraction from SRAM indicates that SRAM PUFs are vulnerable to TLS as well. Recent generations of Intel and Microsemi FPGAs are using SRAM PUFs on their devices for key wrapping and authentication purposes, respectively. Thus, in principle, the attacker can employ TLS to read out the start-up values of the SRAM, which constitute the responses of the PUF. However, since the SRAM PUF is active for only a short amount of time during FPGA configuration and zeroized after generation of the response, the attacker has to halt the IC by some means. For instance, in [NSHB13] the supply voltage of a high-security Infineon SLE66PE smartcard was lowered to the point where its internal clock did not run anymore. At the same time, however, the voltage was high enough to supply the SRAM of the device and acquire stimulation responses from it. Although in this case [NSHB13] no actual data extraction was performed, it can be expected that with a similar approach an attacker would have time to extract the contents of an SRAM PUF by TLS.

6.4 Cell Size and Resolution

The BBRAM memory cell size (approx. 2.8 μm x 3.1 μm) was bigger than expected for a 20 nm device (around 0.3 μm x 0.3 μm based on [CCC+13]). This is probably due to reliability, leakage, and low current consumption considerations. As the key is vital for device operation, designers probably opt for a larger but more reliable memory cell which also delivers a longer battery life. Because of this aspect, it can be expected that BBRAM key storage solutions on other devices and from different vendors exhibit the same vulnerabilities. Although BBRAM can be expected to shrink in the future, if the ratio between a BBRAM and a standard SRAM cell is kept the same, the attack should also work on smaller technology nodes. For example, if a recent solid immersions lens (SIL) option is equipped on the Phemos, the larger numerical aperture (NA) leads to a resolution improvement of $3.1/0.71 = 4.3$ [Ham15]. Assuming the ratio between BBRAM and technology size is kept, the attack can then be expected to work down to the 5 nm node. Yet, it is more meaningful to discuss the resolution limitation in terms of actual

structural cell size and not technology node. For our setup, we expect the TLS data extraction from SRAM to work down to roughly 2 μm cell dimensions and around 600 nm if we employ our SIL. For a more recent SIL, we would expect roughly 470 nm. However, it should be taken into account that the switch to FinFET and similar technologies might bring changes in the behavior of the stimulated cells. It should also be mentioned that larger laser power, smaller spot size, and smaller thermal conductivity lead to a larger temperature gradient. This would lead to a larger TLS signal, but it would not influence the resolution limit. From the used maximum laser power of 21 mW, it can be seen that the total dissipated heat is negligible. Yet, the locally created temperature needs to be kept below the damage threshold of the DUT. For a detailed discussion of these aspects see [BHNF13].

6.5 TLS Availability and Cost

In the scenario presented so far, the attacker is able to rent the necessary equipment by the hour at an FA lab or similar facility such as an university. Considering that the complete setup used in this work is available for rent at $300/hour including operator, this would put the cost for developing the attack at $2100. The cost of extraction of a single device key would come to about $75. Even assuming a time overhead of 100%, this would put the respective cost at $4200 for development and $150 per extraction. However, we should also consider an attacker's options if she does not have access to a ready-to-use TLS setup.

One option is to find a facility with a 1.3 μm (or similar wavelength) laser scanning microscope (LSM) not equipped for TLS and add stimulation capabilities herself. For this, she would simply acquire a current preamplifier, bring it to the lab and connect it to the LSM instead of the reflected light detector signal. Assuming she acquires the amplifier used in this work, this would require her to spend additional $2595 (U.S. list price) [Sys15]. The cost of acquiring the amplifier might be mitigated by the fact that for a simpler setup lower hourly rates can be expected. If she cannot get access to an LSM system at all, she might consider acquiring the whole system. However, this might present her significant costs. Simple LSMs can be expected to cost on the order of tens of thousands of dollars while specialized FA LSMs cost hundreds of thousands of dollars.

For her final option, it should be kept in mind that the demonstrated setup is rather simple and she does not need most of the features of specialized FA equipment. In fact, the only thing required is to move a focused laser spot of a suitable wavelength for thermal interaction across the device while measuring the current. In principle, this can be realized by herself using a laser diode, a microscope objective lens, a mechanical X/Y stage and a current amplifier. Although mechanical stages will require longer acquisition times, they will be less expensive than a complete LSM setup with scanning mirrors and optics. Similarly, she might also be able to use commercially available mechanically scanning setups and have them equipped with a suitable laser. One example of these are systems used for fault injection evaluation, which cost on the order of $50k to $100k depending on options. Correspondence with manufacturers revealed that these can actually be equipped with a 1.3 μm laser for about $20k. As a result, it can be expected that an attacker can either acquire or build a mechanically scanning TLS system for a cost on the order of tens of thousands of dollars, if renting an LSM is not an option.

7 Countermeasure

In this section, we first discuss potential countermeasures to understand whether they are useful against TLS or not. Afterward, we propose a noise-based countermeasure and assess its practicality and effectiveness against TLS by using a prototype circuit in conjunction with the FPGA device.

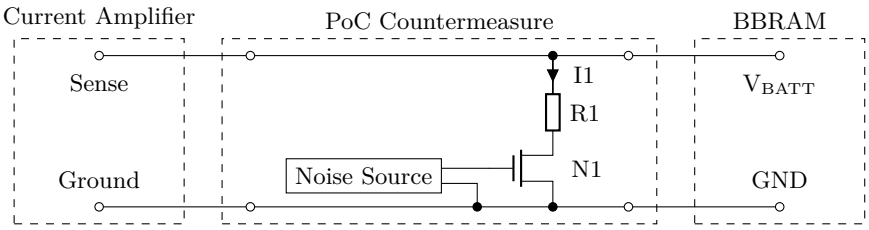


Figure 17: Test setup for the proof-of-concept countermeasure.

At first glance, it might seem that storing the secret key in the eFuses of FPGAs, instead of the BBRAM, increases security. This is due to the fact that the eFuse links themselves can be expected to be mostly unresponsive during laser stimulation. However, it is unknown if stimulation attacks on the circuits connected to the eFuse links might still be possible and reveal the key. Regardless of the inflexibility of using eFuses as the key storage, an adversary can still thin the silicon and read out eFuses by conducting SEM [TM14]. Another option would be storing the key in flash cells, which are more robust against optical attacks. However, this option is not available on SRAM-based FPGAs. An alternative could be the obfuscation of the memory structure itself, for example by distributing the SRAM cells across the chip. Yet, it is unclear if this introduces signal transmission or reliability issues and if it is compatible with current design tools. As discussed in Sect. 6.1, the available obfuscation of the secret key by encryption can in itself only prevent reverse-engineering and not cloning. It furthermore relies on the secrecy of the metalized key, which might be extracted in the future. Also, as the output of decryption circuits can be found in a reasonable amount of time [TLSB17], encryption of the key memory simply adds another step to the attack, but no insurmountable hurdle. A similar situation occurs when scrambling of the key bits is applied. As the influence of individual bits can be directly observed through TLS, they can be quickly mapped. Using a divide-and-conquer approach, only nine TLS measurements are needed to map a 256-bit key. Hence, a better solution is required to safeguard the key storage.

To avert optical attacks in general, a few solutions have been proposed in the literature. One possible countermeasure can be the deployment of silicon light sensors to detect the photons of the laser beam. However, in TLS-based attacks, a wavelength longer than the silicon band gap is utilized. Since no electron-hole pairs are generated by the laser photons, a silicon light sensor is unlikely to trigger. Therefore, to detect a laser beam with such a wavelength, temperature sensors might be useful. However, temperature sensors or temperature sensitive circuits [TFL+17], such as ring-oscillators, have high power consumptions, and thus, they can be active only when the device is powered. Another solution could be having opaque coating layers on the silicon substrate, to obstruct the optical path between transistors and microscope objectives. Since these layers are easily removable by an adversary, active coating layers, which make use of interactions between the protection structure and transistors on the chip, have been introduced to detect removal [AMSB17]. This countermeasure still has a high power consumption and is not protective when the chip is powered off. It is thus useless against the proposed attack.

From this discussion, it can be seen that a TLS-specific countermeasure needs to be developed. There are a few requirements for an active countermeasure: First, the protection scheme should be able to prevent the attack, even when the FPGA is turned off. Hence, it needs to be supplied ideally by the same battery as the BBRAM. In connection with this, it should not drain the backup battery excessively, so that the device can be in its powered-off state for a long time. Second, the countermeasure should be realizable by standard processes and not require any extra manufacturing steps to keep the cost low. Having these requirements in mind, we propose a low-power circuit to make the TLS

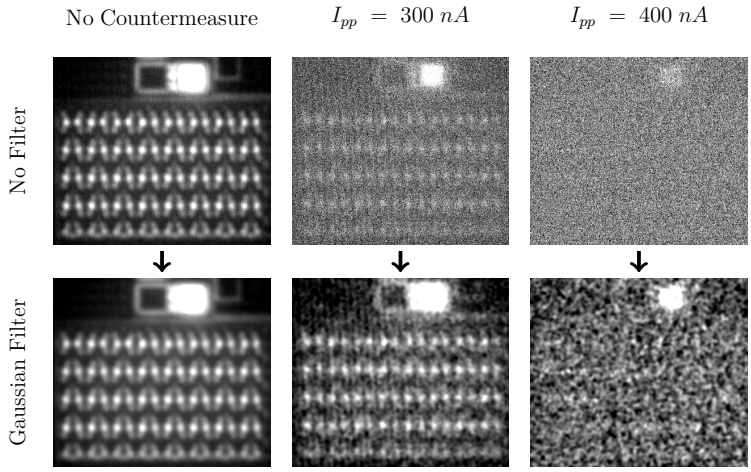


Figure 18: Effect of our proof-of-concept countermeasure on TLS results with and without 2D Gaussian filtering and enhanced contrast applied. Left column: countermeasure disabled, 40% laser power, 72 s scan time. Middle column: countermeasure set at $I_{pp} = 300\ nA$. 100% laser power, 5x120 s scan time. Right column: countermeasure set at $I_{pp} = 400\ nA$. 100% laser power, 5x120 s scan time.

attack more challenging.

By stimulating the SRAM cells with a laser beam in a TLS-based attack, the resulting current fluctuations on the V_{BATT} line can be measured to obtain the key. However, if we have a flow of noisy current on the V_{BATT} line, which masks the data-dependent current leakage, the key cannot be extracted anymore. On the one hand, the noisy current has to have a higher amplitude than the data-dependent leakage current. On the other hand, the source of noisy current should not consume so much power that the battery lifetime decreases. Moreover, by stimulating structures with different geometries on the chip, a data-dependent leakage current with a wideband frequency spectrum is generated. Therefore, ideally, a white noise signal on the V_{BATT} line should mask the leakage current over a wide frequency spectrum.

To generate a masking current on the V_{BATT} line, a noisy current sink can be connected in parallel to the battery. For evaluation, we have devised the proof-of-concept (PoC) countermeasure circuit and test setup presented in Fig. 17. Here, an NMOS transistor N_1 supplied by V_{BATT} with its gate connected to a suitable white noise source is employed. The current I_1 flowing through the transistor N_1 and limiting resistor R_1 should mask the data-dependent current leakage on V_{BATT}. Note that I_1 can be characterized by its mean or offset current I_{mean} and the amplitude of the fluctuations around I_{mean} referred to here as I_{pp}. By selecting a suitable resistor R_1, the average current I_{mean} can be set low enough to not draw too much power from the battery. At the same time, a suitable I_{pp} fluctuation can be chosen to mask the data-dependent leakage current. However, I_{pp} cannot be chosen arbitrarily high, as for symmetric fluctuations $I_{pp\ max} = 2I_{mean}$ in the proposed circuit. For an actual circuit integrated into the FPGA, the white noise source can be realized by electronic components, such as Zener diodes, and amplified to drive and modulate the gate of N_1. However, in the PoC experiments, we employed a Keithley 3390 function generator in "noise" mode. As seen in Fig. 17, our PoC circuit is connected in parallel to the current amplifier and the BBRAM. In the case of an actual implementation, the circuit would naturally be located on the FPGA die to prevent tampering.

The effect of the countermeasure circuit on TLS measurements can be observed in Fig. 18, where only the left block of the BBRAM is shown. First, the countermeasure is

disabled, and a TLS measurement is carried out with 40% laser power and 72 s scan time. As expected, the stored values in BBRAM can be clearly decoded, as can be seen in the first column of Fig. 18. Application of a $2D$ Gaussian smoothing filter, popular in image processing to remove noise, further improves the measurement result. Next, we enabled our countermeasure by generating a noise current with $I_{mean} = 1.3\ \mu A$ and $I_{pp} = 300\ nA$. This forced us to decrease the current amplification to 1 μA/V to prevent overloading of the preamplifier input. As no signal could be acquired using the original settings, we increased laser power to 100%, scan time to 120 s and averaged five measurements to reduce the effect of the countermeasure. However, as evident in the second column of Fig. 18, the signal-to-noise ratio (SNR) of the obtained TLS signal is still reduced drastically. Although applying the Gaussian filter increases the SNR of the captured signal, not all stored bits can be decoded clearly. Finally, we increased the peak-to-peak amplitude of the noise current to $I_{pp} = 400\ nA$. As a result, the SNR is decreased to a point where not even the general BBRAM structure can be detected, regardless of whether Gaussian filtering is used or not.

Although our countermeasure is only a PoC circuit, it is worth considering what battery lifetime it could achieve. The minimum recommended V_{BATT} voltage is 1 V, on which we add a safety margin of 20% [Xil18a]. At 1.2 V discharge, the capacity of the battery supplied with the Ultrascale board is 140 mAh [Ene18]. As our countermeasure draws an average of 1.3 μA and the BBRAM is rated at 150 nA maximum [Xil18a], the total draw is 1.45 μA. It should be noted, that we omit the average current consumption of the noise source here, as we currently have no way of estimating it. If the system was *permanently* disconnected from *all* power sources, this would still lead to a key retention of eleven years. As soon as power is available to the FPGA, the battery is automatically disconnected. Thus, during normal operation, the battery can be expected to fail from old age before it is drained. As a consequence, although our approach is only a PoC, it does not reduce effective battery life. Even if a future implementation would reduce battery life, it can still be considered as a trade-off in high-security scenarios.

This demonstrates that there is much room for the power consumption of the noise source and that the countermeasure could be designed much more aggressively and still achieve a satisfying battery lifetime. Furthermore, the I_{pp} fluctuations of our PoC circuit are not optimal at this point, as they could in principle be much larger while achieving the same I_{mean}. Additionally, more advanced countermeasures based on the same principle, which do not increase current draw at all, are possible. For example, if a capacitor is randomly connected with alternating polarity between V_{BATT} and ground, it will charge from and discharge into the V_{BATT} net. This will create noisy currents, but, omitting parasitic resistance, will not influence the average current draw, as it is sinking *and sourcing* current. Thus, it can be seen that there is room for the development of low-power current noise sources as TLS countermeasures. However, the basic effectiveness of such an approach has been proven.

8 Conclusion

IC vendors have integrated several protection schemes into their products to counteract physical attacks. However, while a wide variety of attack categories (e.g., DPA) have been taken seriously by the industry in the recent years, the threat of optical attacks from the IC backside has been ignored. The presented experiments in this work confirmed again that an optical attack, such as TLS, is indeed a powerful attack technique for key extraction from real-world devices even on latest technology nodes. It became further apparent that TLS can be mounted in a very short amount of time even if little or no knowledge about the underlying hardware design is available to the adversary. Moreover, the lower cost and higher availability of TLS in comparison to other optical attacks, such as optical contactless

probing, makes this technique even more threatening. We demonstrated that the stored key in the BBRAM of the FPGA can be extracted when the FPGA is disconnected from power. Hence, conventional countermeasures are incapable of preventing such an attack. Consequently, we proposed, constructed and demonstrated a low-power countermeasure scheme, which can be supplied by a battery when the device is turned off. Since the proposed countermeasure is compatible with CMOS technology, it is feasible to integrate it into the chip at low-cost.

References

[AMSB17] Elham Amini, Ruslan Muydinov, Bernd Szyszka, and Christian Boit. Backside Protection Structure for Security Sensitive ICs. In *43rd International Symposium for Testing and Failure Analysis (November 5-9, 2017)*. Asm, 2017.

[AVN18] AVNET. Xilinx Kintex UltraScale Development Kit. https://www.xilinx.com/products/boards-and-kits/1-buuhej.html, 2018.

[BDPL01] F Beaudoin, R Desplats, P Perdu, and D Lewis. Implementing thermal laser stimulation in a failure analysis laboratory. In *International Symposium for Testing and Failure Analysis*, pages 151–160. ASM International, 2001.

[BHNF13] Christian Boit, Clemens Helfmeier, Dmitry Nedospasov, and Alexander Fox. Ultra high precision circuit diagnosis through seebeck generation and charge monitoring. In *Physical and Failure Analysis of Integrated Circuits (IPFA), 2013 20th IEEE International Symposium on the*, pages 17–21. IEEE, 2013.

[CCC+13] Jonathan Chang, Yen-Huei Chen, Hank Cheng, Wei-Min Chan, Hung-Jen Liao, Quincy Li, Stanley Chang, Sreedhar Natarajan, Robin Lee, Ping-Wei Wang, et al. A 20nm 112Mb SRAM in High-K metal-gate with assist circuitry for low-leakage and low-V MIN applications. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2013 IEEE International*, pages 316–317. IEEE, 2013.

[Ene18] Energizer. Energizer A76 - Product Datasheet. http://data.energizer.com/pdfs/a76z.pdf, 2018.

[Ham15] Hamamatsu. NanoLens-SHR. https://www.hamamatsu.com/resources/pdf/sys/SSMS0053E_Nanolens-SHR.pdf, 2015.

[Int17] Intel. AN 556: Using the Design Security Features in Intel FPGAs. https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/an/an556.pdf, 2017.

[Its18] Itseez. Open Source Computer Vision Library. https://github.com/itseez/opencv, 2018.

[LKA18] Ting Lu, Ryan Kenny, and Sean Atsatt. White Paper: Secure Device Manager for Intel® Stratix® 10 Devices Provides FPGA and SoC Security. *Intel, Santa Clara, CA*, 2018.

[Low04] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. In *International Journal of Computer Vision*, volume 60, pages 91–110, 2004.

[MS16] Amir Moradi and Tobias Schneider. Improved Side-Channel Analysis Attacks on Xilinx Bitstream Encryption of 5, 6, and 7 Series. In *Constructive Side-Channel Analysis and Secure Design – COSADE 2016*. Springer, 2016.

[NSHB13] Dmitry Nedospasov, Jean-Pierre Seifert, Clemens Helfmeier, and Christian Boit. Invasive PUF Analysis. In *Fault Diagnosis and Tolerance in Cryptography (FDTC), 2013 Workshop on*, pages 30–38. IEEE, 2013.

[Ots79] N. Otsu. A Threshold Selection Method from Gray-Level Histograms. In *IEEE Transactions on Systems, Man and Cybernetics*, volume 9, pages 62–66, 1979.

[Pet17] Ed Peterson. XAPP1098: Developing Tamper-Resistant Designs with UltraScale and UltraScale+ FPGAs. *Xilinx, Inc. San Jose, CA*, 2017.

[PST09] Stephan Preibisch, Stephan Saalfeld, and Pavel Tomancak. Globally Optimal Stitching of Tiled 3D Microscopic Image Acquisitions. volume 25, pages 1463–1465. Oxford Univ Press, 2009.

[SACF+12] Johannes Schindelin, Ignacio Arganda-Carreras, Erwin Frise, Verena Kaynig, Mark Longair, Tobias Pietzsch, Stephan Preibisch, Curtis Rueden, Stephan Saalfeld, and Benjamin Schmid. Fiji: An Open-Source Platform for Biological-Image Analysis. In *Nature methods*, volume 9, page 676, 2012.

[Sko06] Sergei Skorobogatov. Optically enhanced position-locked power analysis. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 61–75. Springer, 2006.

[SMOP15] Pawel Swierczynski, Amir Moradi, David Oswald, and Christof Paar. Physical Security Evaluation of the Bitstream Encryption mechanism of Altera Stratix II and Stratix III FPGAs. volume 7, page 34. ACM, 2015.

[SSAQ02] David Samyde, Sergei Skorobogatov, Ross Anderson, and J-J Quisquater. On a new way to read data from memory. In *Security in Storage Workshop, 2002. Proceedings. First International IEEE*, pages 65–69. IEEE, 2002.

[Sys15] Stanford Research Systems. Low-Noise Current Preamplifier. http://www.thinksrs.com/downloads/PDFs/Catalog/SR570c.pdf, 2015.

[TFL+17] Shahin Tajik, Julian Fietkau, Heiko Lohrke, Jean-Pierre Seifert, and Christian Boit. PUFMon: Security Monitoring of FPGAs using Physically Unclonable Functions. IEEE, 2017.

[TGB17] Mark M. Tehranipoor, Ujjwal Guin, and Swarup Bhunia. Invasion of the Hardware Snatchers: Cloned Electronics Pollute the Market. 2017.

[TLSB17] Shahin Tajik, Heiko Lohrke, Jean-Pierre Seifert, and Christian Boit. 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*, CCS '17, pages 1661–1674, New York, NY, USA, 2017. ACM.

[TM14] Stephen M Trimberger and Jason J Moore. FPGA Security: Motivations, Features, and Applications. *Proceedings of the IEEE*, 102(8):1248–1265, 2014.

[VT13] John Villasenor and Mohammad Tehranipoor. The Hidden Dangers of Chop-Shop Electronics: Clever Counterfeiters Sell Old Components as New Threatening both Military and Commercial Systems. 2013.

[Wil17] Kyle Wilkinson. Using Encryption and Authentication to Secure an UltraScale/UltraScale+ FPGA Bitstream. *Xilinx, Inc. San Jose, CA*, 2017.

[Xil12] Xilinx. Vivado Design Suite User Guide. https://www.xilinx.com/support/documentation/sw_manuals/xilinx2012_2/ug893-vivado-ide.pdf, 2012.

[Xil18a] Xilinx. Kintex UltraScale FPGAs Data Sheet: DC and AC Switching Characteristics. https://www.xilinx.com/support/documentation/data_sheets/ds892-kintex-ultrascale-data-sheet.pdf, 2018.

[Xil18b] Xilinx. UltraScale Architecture Configuration - User Guide. https://www.xilinx.com/support/documentation/user_guides/ug570-ultrascale-configuration.pdf, 2018.

[Xil18c] Xilinx. Xilinx Embedded Software (embeddedsw) Development. https://github.com/Xilinx/embeddedsw, 2018.