Testing Feasibility of Back-Side Laser Fault Injection on a Microcontroller

Jakub Breier[1] **and Dirmanto Jap**[2]
Physical Analysis and Cryptographic Engineering
Temasek Laboratories@NTU[1]
School of Physical and Mathematical Sciences[2]
Nanyang Technological University, Singapore
jbreier@ntu.edu.sg, dirm0002@e.ntu.edu.sg

Abstract

Laser fault attack platform constitutes a powerful tool for a precise injection of faults into the device, allowing an attacker to carefully adjust timing and position on the chip. On the other hand, the cost of such equipment is high and the profiling time is non-negligible.

In this paper, we would like to investigate the practicability of the back-side laser fault injection and to state benefits and drawbacks of this technique. We performed experiments on two methods of fault injections induced by a laser beam – instruction disturbance experiments and register value changes. The first method, as our experiments show, is easy to perform, precise and repeatable. The second one is harder to perform and we could not achieve repeatability in such experiments.

Keywords: Laser, Fault Attack, ATmega328P

1 Introduction

To perform an attack on cryptographic implementation on an integrated circuit, the attacker could either target the theoretical flaw or weakness in the design (classical cryptanalysis), the physical leakage resulting from the implementation (side-channel analysis) or by reverse engineering the device (invasive attack).

An alternative is the fault attack, where the fault is injected to affect the internal state when executing the cryptographic algorithm. By observing the faulty behavior, the attacker could obtain some information regarding the secret key or plaintext. There are different methods of injecting faults, but here, we would focus on fault injection using the laser (Light Amplification by Stimulated Emission of Radiation) beam.

The possibility to insert a fault in the algorithm execution for the purpose of revealing the secret key from the device was introduced by Boneh, DeMillo

and Lipton in 1996 [3].

The first practical attack using the real device was implemented by Biham and Shamir [2]. They used a technique similar to Differential Cryptanalysis, called Differential Fault Analysis (DFA). This technique exploits a possibility to insert a fault into last rounds of an algorithm execution, which results into a faulty ciphertext. Attacker then compares this ciphertext with the original one and gains information about the secret key. DFA is the most popular fault attack technique for attacking symmetric ciphers.

There are several other techniques for fault injection attacks, which can be used for revealing the key. Except DFA, the popular techniques are: Collision Fault Analysis (CFA), Ineffective Fault Analysis (IFA), and Safe-Error Analysis (SEA) [5, 8].

In general, most of the presented attacks are only verified theoretically through simulation. In most cases, these attacks require a strong assumption on the fault model. This paper provides experimental tests on a laser fault injection station we have performed in order to examine the practicability of different laser fault injection techniques.

We performed our experiments on the Atmel ATmega328P microcontroller, which is one of the most common microcontrollers available on the market due to the popular Arduino UNO platform. We examined two methods of laser fault injections – an instruction disturbance attack and a register value change. While the first one is easy to perform and reproduce, and the area can be localized fast, the second one cannot be repeated easily with same faults, the laser beam power has to be very strong, and most of the time, bytes affected cannot be determined with high success rate.

This paper is structured as follows: Section 2 provides an overview of the most significant works in the area of laser fault injections. Next, we provide a brief theoretical background on the laser fault injection in Section 3, followed by our experimental setup in Section 4 and the results in Section 5. Finally, we provide a discussion of the results in Section 6 and in Section 7, we conclude our experiments.

2 Previous Works

A practical laser fault injection attacks have been successfully implemented before. However, majority of the presented attacks is based on random fault models, which could be easily achieved without the need of a laser injection. In this section, we will briefly describe papers concerning more advanced laser fault injection experiments.

In [6], Courbon et al. reported practical application of a back-side laser fault injection on a 90 nm microcontroller. They managed to set/reset a byte stored in the register. They concluded that bit resets can be done with lower energy than the energy needed for bit sets. Later [7], they successfully conducted an attack on AES implementation running on a 130 nm microcontroller. In this experiment, they have opened another chip from the front-side in order to

identify the flip flops using a Scanning Electron Microscope. After determining these potential points of interest, they were able to narrow the area which had to be scanned by the laser to perform a successful attack on registers. However, there is no detailed report on the experimental setup, and it can be safely assumed that producing such results needs very precise and expensive equipment.

Agoyan et al. [1] presented a DFA attack on AES implemented on a 350 nm microcontroller by performing multiple byte faults. They aimed at the SRAM, attacking the surface of the chip from the front side. They used a green laser beam ($\sim$ 532 nm) with 5.5 μm diameter, 20x magnifying objective lens and a positioning table with 0.1 μm precision.

Roscian et al. [11] had also investigated possibilities of a laser fault injection, using AES implementation running on ASIC. They used relatively large laser beam spot (square spot 125x125 μm^2) and they could successfully perform bit flips and bit sets/resets. Despite the size of the beam, a large part of induced faults were single-bit faults. With these results they were able to perform a successful DFA on AES.

3 Laser Fault Injection

There are two ways to inject faults into an integrated circuit using a laser beam:

- **Front side attacks** - green (532 nm) and red (808 nm) lasers can be used for these attacks. Visibility of components makes these attacks easier. Reflective effect of the metallic components can lower the accuracy. Also, bonding wires could be cut in the process of de-packaging, which would make the chip useless.
- **Back side attacks** - near infrared (1064 nm) laser is suitable for these attacks, because laser needs to go through the silicon layer. Positioning is harder because the components are not visible but there is no problem with the reflection.

There are two essential parameters when considering back-side laser fault injection – the *reflection coefficient* and the *absorption coefficient*. The *absorption coefficient* determines how far can a light with a certain wavelength penetrate into a particular material before it is completely absorbed. The *reflection coefficient* is a ratio of the amplitude of the reflected wave and the amplitude of the incident wave.

If we consider the peak laser intensity I_p, thickness of a silicon substrate on the back side of the chip d, reflection coefficient R and optical absorption of a silicon α, we can compute optical properties by using Equation 1 [10]. I_0 is the fraction of an incident intensity transmitted by the backside interface.

$$I_p = I_0(1 - R)e^{-\alpha d} \tag{1}$$

Figure 1 shows reflectivity of a polished silicon wafer with respect to particular wavelengths. In Figure 2 we can see the absorption depth, computed as an

inverse of the *absorption coefficient*. This plot clearly shows us why we cannot use green (532 nm) or red (808 nm) laser beams for back side fault injection attacks, since the green laser can penetrate ~ 1 μm thick silicon substrate and the red laser can penetrate ~ 10 μm thick silicon substrate.

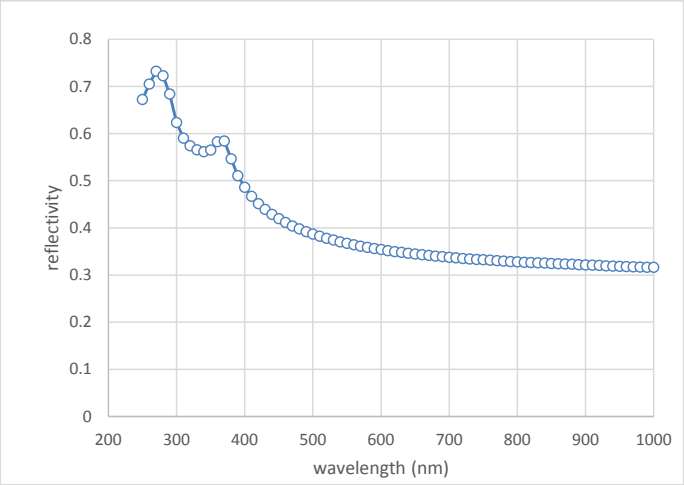

Figure 1: Reflectivity of Silicon for particular wavelengths [9].

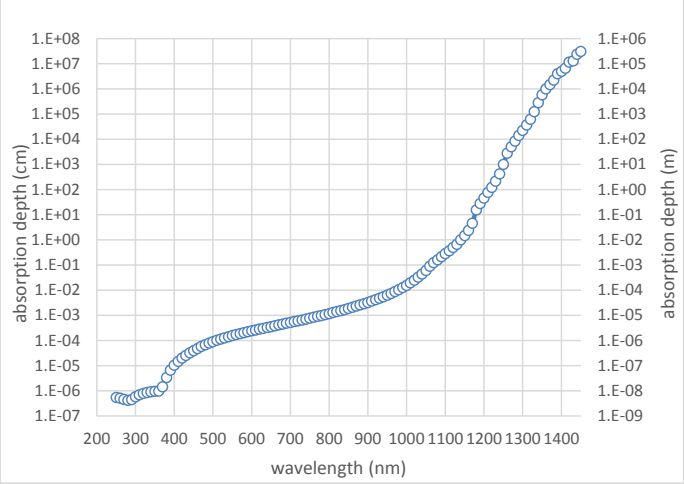

Figure 2: Absorption depth for particular wavelengths in Silicon [9].

The advantages of using laser to perform fault attacks are claimed to be its precision and reproducibility. Spatial coherence allows a laser beam to be focused to a tight spot, therefore with a small diameter it is possible to aim at very small components of an integrated circuit. With focused beam it is be possible to disturb transistors and change bit values in registers. Also, given

the same parameters, it should be possible to repeat the experiment with high probability of the same results.

However, there are also some disadvantages connected with a usage of a laser equipment for the fault injection. First, the chip surface has to be accessible by the laser, which means that we need to de-package the chip before. This phase could be done using different methods, either by using chemicals or by physically grinding and milling the epoxy package until the chip is visible. Another disadvantage is a possibility of destroying the chip either by a large number of repetitions or by a high laser energy.

4 Experimental Setup

The main part of our setup is the near infrared diode pulse laser, having following properties:

- Pulse power: 20 W (reduced to 8 W with 20x objective and 7 W with 50x objective)
- Pulse repetition: 10 MHz
- Spot size: 30x12 μm^2 (15x3.5 μm^2 with 20x objective and 6x1.4 μm^2 with 50x objective)
- Response to trigger pulse: $\leq$ 60 ns

As a device under test (DUT), we used 8-bit Atmel ATmega328P microcontroller. This microcontroller is running at 16 MHz, it has 1 KB EEPROM, 32 KB of Flash memory, and 2 KB SRAM. It was produced by using 350 nm manufacturing process. There is a trigger signal set on HIGH (5 V) during the algorithm execution in order to identify the correct time for the fault injection. The area of the chip is 3x3 mm^2 large.

The DUT is mounted on the Arduino UNO development board. Since the DUT is de-packaged from the back side, we have bent the connector pins and soldered an additional socket on the back side of the board. The board is then mounted on the X-Y positioning table with a step precision 0.05 μm. This setup is depicted in Figure 3.

Communication with the DUT is done via RS232 interface. We used an oscilloscope for measuring the power consumption of the DUT, for capturing the trigger signal and the laser diode current, so we could determine the delay between sending the trigger signal and activating the laser beam.

5 Results

For our experiments, we first scanned the whole area of the chip using random input, and recorded the location where faulty outputs were observed. Then, more detailed explorations were performed on that location.

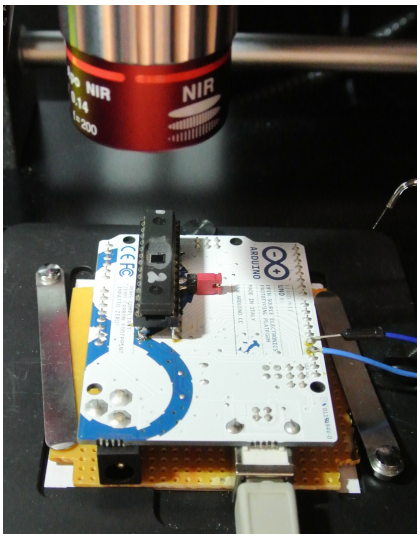

Figure 3: Device under test during the experiment.

Code for our experiments was written in assembly language, using the Arduino programming platform (code snippet is stated in Table 5). Experiment steps for the register disturbance were following:

1. Send 10 bytes to the device via RS232 interface from the PC.
2. Store the data in a variable.
3. Load each byte to a different register.
4. Set the trigger signal on Arduino pin 13 to HIGH (5V).
5. Perform 10 `nop` instructions (10x62.5 ns). This step is crucial in order to avoid fault injection in the bus or in the clock signal. It prevents executing instructions during the laser beam activation.
6. Activate the laser beam.
7. Read the data from registers and send them back to the PC.
8. Compare the data.

For instruction disturbance experiments the steps were similar, only the trigger signal was sent between steps 2 and 3.

5.1 Instruction Skip Experiments

First, we used 20x magnification lens, and we observed that with a small laser power (0.5-0.9%) it was enough to disturb the instruction execution on the

Table 1: Snippet of assembly code used in the DUT.

Instruction	Description
LDI r0, 0x20	set trigger
LD rN,X+	(x10) load the value to different registers
OUT 5, r0	send trigger signal
NOP	(x10) execute no operation
ST Y+, rN	(x10) send the value from different registers to a variable

chip and to perform an instruction skip. As mentioned earlier, we performed preliminary experiments and we fixed a specific location on the chip. We varied parameters of the laser setup, such as laser strength, glitch length and glitch offset. Our experimental results showed that it is possible to precisely determine the position of instruction being executed and to skip the loading to a register so that the value obtained is shifted.

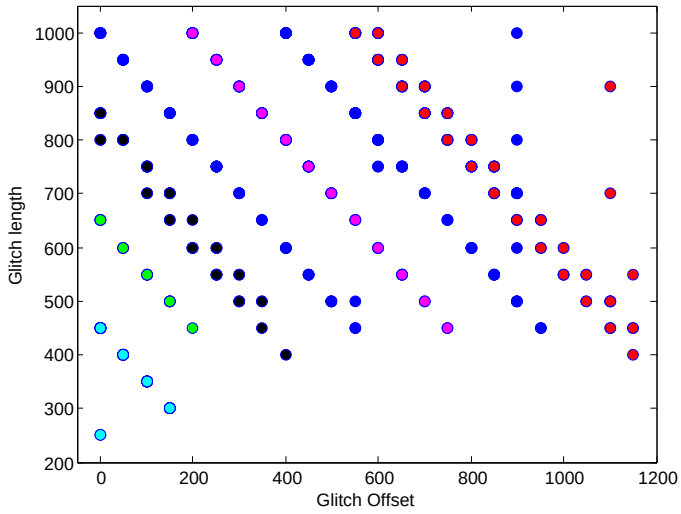

Figure 4: Skip instruction faults with different laser beam parameters. Colors represent faults on different bytes.

We observed that there is a relation between the sum of the glitch length, the offset and a byte affected (shown by different colors in Figure 4). This corresponds to the order of instructions execution, such that the lower byte will be executed later than the higher byte, and the sum of the glitch length and the offset indicates the point in time where the instruction is executed. Another observation is that the data is not affected uniformly, some bytes tend to be affected more frequently than others, as could be seen in Figure 5. With

increasing laser power we have not observed any other types of faults in the same area, we have only increased the probability of a successful fault injection.

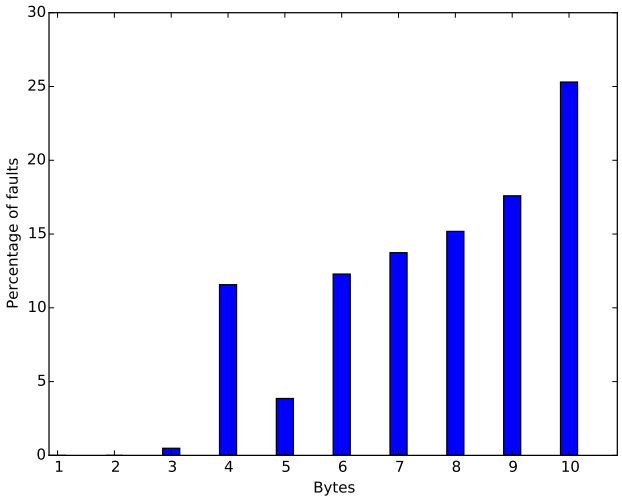

Figure 5: Proportion of faults for different bytes.

5.2 Register Disturbance Experiments

In order to affect the value in registers, we had to increase the laser power significantly. Starting with 19% power, using 20x magnification objective without any filter, we managed to disrupt one byte in a register. For the preliminary experiments, we varied the length of the laser glitch between 10-2000 ns. We observed that starting from approximately 100 ns, the faulty output of one byte (byte 6 in register `r20`) was obtained, and thus we set the glitch length to 150 ns. In Figure 6, we observed there is a pattern regarding different faults we obtained. The square corresponds to an input value `0xFF` and the circle corresponds to an input value `0x00`. The rectangular pattern is due to the shape of the laser spot. However, as we further increased the strength of the laser up to 100% with 150 ns glitch length, we could not obtain faults on any other bytes. The most probable reason of this behavior is a large beam spot that is unable to aim at particular registers.

After swapping the lens to 50x magnification, and increasing the laser power up to 34%, we observed no faulty outputs. This might be due to the precision of the laser, as the laser beam focusing is more difficult than with 20x magnifying lens. Also, the resulting laser power is slightly reduced when switching from 20x lens to 50x lens (8W to 7W). With 35% power we managed to obtain several faults which helped us to isolate the targeted area on the chip. We used three types of inputs, random input, `0x00` input and `0xFF` input. Laser power for

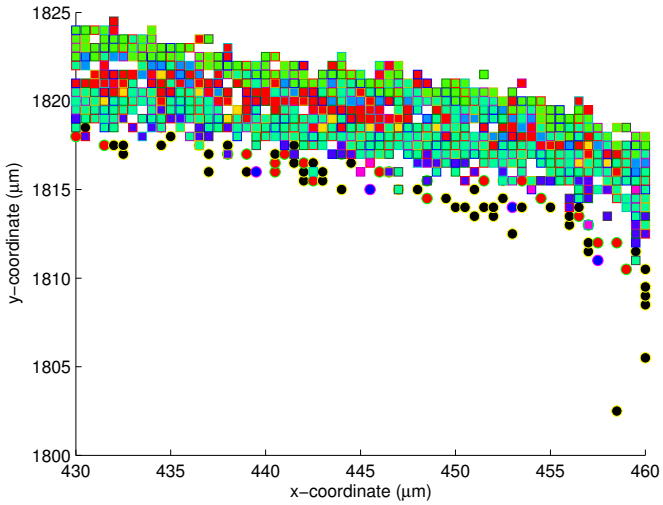

Figure 6: Faults on different locations on the chip.

these experiments was set to 60% and glitch length to 150 ns. The faulty area can be seen in figure 7. In this area, faults were divided into two categories:

1. **Stuck-at faults:** For stuck-at faults, either value in the first or second byte was changed to one of the following values: {`0x45,0x85,0xCA,0xC5,0xE5,0xFA`}. In this case, fault was not dependent on the input value. These faults are depicted by blue color.

2. **Repeating previous value:** In 17.15% of all faulty cases, the output took a value of the previous output, no matter if that value was correct or faulty with respect to the previous input. In this case we can assume that one of the registers (`r28, r29`, which hold memory address for variable `Y` (where the output is supposed to be written before sending the data back to PC), was changed. Therefore the data is written to another address in memory and old value remains in `Y`. These faults are depicted by red color in the picture.

We also calculated the repeatability of these faults. We repeated the experiments with fixed parameters and location 5000 times, for all three types of inputs. The chances of repeating the experiments with `0x00` input was 56%, for `0xFF`, the chance was 39.5%, and for the random input, it was only 25.6%.

5.3 Address Change Faults

We have observed another faulty behavior of the microcontroller – changing the address in an instruction. For this experiment we slightly changed the testing

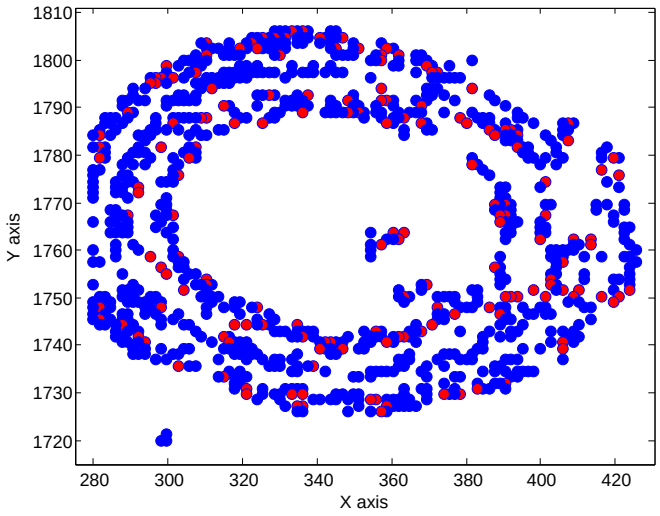

Figure 7: Register value changes – faulty area.

procedure to send and receive 25 random bytes, so that all the registers are used. This type of faulty behavior was observed by using 25% laser power and 120 ns glitch length.

In each experiment, an address of the last register was changed in the `LOAD` instruction. Figure 8 shows the situation when the last register was `r25`. Similar situation was observed when other registers were used. Faults were dependent on the area of the microcontroller. Colors represent following behavior:

- **Green:** value of `r25` was set to `0x00`.
- **Blue:** address was changed to load the data from `r9` (address bits changed from 11001 to 01001).
- **Yellow:** address was changed to load the data from `r17` (address bits changed from 11001 to 10001).
- **Magenta:** address was changed to load the data from `r24` (address bits changed from 11001 to 11000).
- **Red:** value was changed in two different bytes, but the faulty behavior remains unknown.

6 Discussion

In our experiments, we were able to disturb the instruction execution on a microcontroller and to change the values in registers.

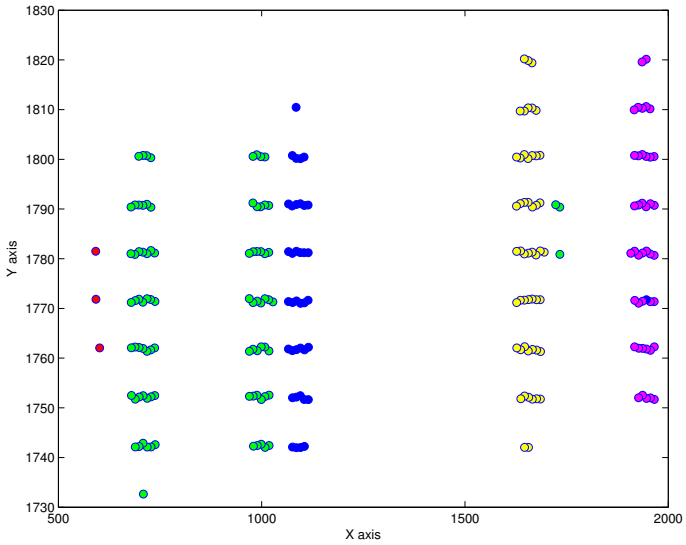

Figure 8: Address change faults.

The skip instruction attack is relatively easy to perform, requires low laser power (0.5 - 0.9%) and the area on the chip can be easily identified with 20x magnifying lens. It was possible to identify the vulnerable area of the chip with relatively large step size (200x200 steps, resulting to 15 μm step size). From the time point of view, the initial localization, analysing 40,000 spots took approximately 24 minutes. This type of experiment is easily repeatable and could be used for any fault attack requiring either a random fault model or skipping of a particular instruction. We could disturb 8 out of 10 bytes, however, in order to attack the last 2 bytes, we needed to make the sum of the glitch length and offset longer. However, if we further increased any of those parameters, the microcontroller stopped responding.

Another instruction disturbance faults resulted into address change. We were able to change the address of the last register in our send/receive testing program. This change was dependent on the area and required higher laser power than the instruction skip (25% power). Again, the success rate was dependent on the correct determination of parameters so that same settings produced same faulty output.

The situation with register value change experiments was different. We had to set the laser power at least to 19% to successfully localize the registers area. After changing the magnifying objective to more precise one (50x), it was necessary to further increase the power to 35%. Further increase did not have any impact on changed values or affected registers, it only prevented microcontroller from performing operations, therefore we had to repeat the experiments. The laser beam spot could only be set to be as precise as 6 x 1.4 μm^2. It was not

possible to affect all the registers. Also, we could not determine the value in registers after the fault injection, since the faulty mask was unpredictable. This type of fault injection could be used to change the byte of an intermediate value, but the attacker would have very weak control over the changes.

We performed experiments on two identical ATmega328P microcontrollers with same results. During the experiments, we could increase the energy of the laser up to few hundred nJ without damaging the chip. Therefore, we can conclude that this microcontroller has a high tolerance against laser fault injection and cannot be easily damaged with a back-side laser radiation.

7 Conclusion

In our paper we performed experiments on attacking the back side of the Atmel ATmega328P microcontroller, using a standard laser fault injection platform. The price of the whole setup is below 150,000 EUR.

We considered two methods for fault injection – instruction disturbance and register value change. The first one can be used for random byte fault attacks and instruction skips, for example, as shown in [4], it can be used to skip `xor` operation used for post-whitening in block ciphers in order to get the last round key. The second one, if performed very precisely, could be used for bit-flip fault models, which allow the most powerful fault attacks on cryptosystems.

We observed that the reproducibility of laser faults when performing register value change faults is very low. As we have shown, we could only get faulty result in approximately 40-50% of cases when fixing all the parameters, and types of obtained faults are different. It is relatively hard to achieve some of the fault models such as bit sets/resets. Most of the results we obtained are random byte faults.

There are some parameters which should be taken into consideration in order to increase the chance of success:

- Thickness of a silicon substrate – as shown in Figure 2, the thickness of a material plays a key role when considering a photon penetration into the silicon.
- Smoothness of a chip surface – if the surface of the back side of a chip is not smooth enough, it can cause refraction and scattering of a laser beam, making a precise attack harder or impossible.
- Beam spot size – the smaller the better. Current manufacturing technologies allow the transistor sizes smaller than 20 nm, therefore it is necessary to have the beam spot size small enough if we want to avoid affecting multiple registers at the same time.
- Precise positioning table – the reason is the same as in the previous case, without a precise positioning it may be impossible to target specific register on a chip.

There are associated some problems with the laser fault injection, which make this attack harder to perform than other fault injection techniques. It is nearly impossible to make a profiling which fits multiple devices, since each chip has a different layout, different manufacturing process and even if we are aiming at the specific chip, de-packaging can cause small but significant differences on the surface which can result in a different fault sensitivity. The second problem is a chip survivability. The microcontroller we used for our experiments was durable enough to withstand several weeks of experiments in a row without any observable damage. This could be due to the old manufacturing process (350 nm), making the connections and transistors large. However, with more advanced technologies, the size of components is much smaller and therefore chips are more vulnerable to optical fault attacks.

References

[1] M. Agoyan, J.-M. Dutertre, A.-P. Mirbaha, D. Naccache, A.-L. Ribotta, and A. Tria. Single-bit DFA using multiple-byte laser fault injection. In *Technologies for Homeland Security (HST), 2010 IEEE International Conference on*, pages 113–119, Nov 2010.

[2] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In J. Kaliski, BurtonS., editor, *Advances in Cryptology - CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer Berlin Heidelberg, 1997.

[3] D. Boneh, R. A. DeMillo, and R. J. Lipton. On the Importance of Checking Cryptographic Protocols for Faults. In *Proceedings of the 16th Annual International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT'97, pages 37–51, Berlin, Heidelberg, 1997. Springer-Verlag.

[4] J. Breier, D. Jap, and C.-N. Chen. Laser Profiling for the Back-Side Fault Attacks: With a Practical Laser Skip Instruction Attack on AES. In *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*, CPSS '15, pages 99–103, New York, NY, USA, 2015. ACM.

[5] C. Clavier. Attacking Block Ciphers. In M. Joye and M. Tunstall, editors, *Fault Analysis in Cryptography*, Information Security and Cryptography, pages 19–35. Springer Berlin Heidelberg, 2012.

[6] F. Courbon, P. Loubet-Moundi, J. Fournier, and A. Tria. Adjusting laser injections for fully controlled faults. In E. Prouff, editor, *Constructive Side-Channel Analysis and Secure Design*, Lecture Notes in Computer Science, pages 229–242. Springer International Publishing, 2014.

[7] F. Courbon, P. Loubet-Moundi, J. Fournier, and A. Tria. Increasing the efficiency of laser fault injections using fast gate level reverse engineering. In

Hardware-Oriented Security and Trust (HOST), 2014 IEEE International Symposium on, pages 60–63, May 2014.

[8] J.-L. Danger, S. Guilley, P. Hoogvorst, C. Murdica, and D. Naccache. A synthesis of side-channel attacks on elliptic curve cryptography in smartcards. *Journal of Cryptographic Engineering*, 3(4):241–265, 2013.

[9] M. A. Green and M. J. Keevers. Optical properties of intrinsic silicon at 300 k. *Progress in Photovoltaics: Research and Applications*, 3(3):189–192, 1995.

[10] D. Lewis, V. Pouget, F. Beaudoin, P. Perdu, H. Lapuyade, P. Fouillat, and A. Touboul. Backside laser testing of ICs for SET sensitivity evaluation. *Nuclear Science, IEEE Transactions on*, 48(6):2193–2201, Dec 2001.

[11] C. Roscian, J.-M. Dutertre, and A. Tria. Frontside laser fault injection on cryptosystems - Application to the AES' last round. In *Hardware-Oriented Security and Trust (HOST), 2013 IEEE International Symposium on*, pages 119–124, June 2013.