Experimental Analysis of the Laser-Induced Instruction Skip Fault Model

Jean-Max Dutertre, Timothé Riom, Olivier Potin, Jean-Baptiste Rigaud

▶ **To cite this version:**

Jean-Max Dutertre, Timothé Riom, Olivier Potin, Jean-Baptiste Rigaud. Experimental Analysis of the Laser-Induced Instruction Skip Fault Model. The 24th Nordic Conference on Secure IT Systems, Nordsec 2019, Nov 2019, Aalborg, Denmark. pp.221-237, 10.1007/978-3-030-35055-0_14 . hal-02379754

HAL Id: hal-02379754
https://hal.science/hal-02379754

Submitted on 25 Nov 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Authors' version

https://doi.org/10.1007/978-3-030-35055-0_14
The 24th Nordic Conference on Secure IT Systems, Nordsec 2019

Experimental Analysis of the Laser-induced Instruction Skip Fault Model

Jean-Max Dutertre, Timothé Riom, Olivier Potin, and Jean-Baptiste Rigaud

Mines Saint-Etienne, CEA-Tech, Centre CMP, F - 13541 Gardanne France,
name@emse.fr

Abstract. Microcontrollers storing valuable data or using security functions are vulnerable to fault injection attacks. Among the various types of faults, instruction skips induced at runtime proved to be effective against identification routines or encryption algorithms. Several research works assessed a fault model that consists in a single instruction skip, i.e. the ability to prevent one chosen instruction in a program from being executed. This assessment is used to design countermeasures able to withstand a single instruction skip. We question this fault model on experimental basis and report the possibility to induce with a laser an arbitrary number of instruction skips. This ability to erase entire sections of a firmware has strong implications regarding the design of countermeasures.

1 Introduction

Fault attacks (FA) consist in disturbing the operations of a target integrated circuit (IC) for the purpose of extracting secret information it may contain. Faults, or computation errors, are injected by means of altering the target environmental conditions [2] (e.g. its voltage, temperature, frequency, etc.). The induced information leakage generally aims at extracting a cryptographic key [4] or at providing an unauthorized access to some of the target functionalities [9].

Laser illumination may also be used to inject faults into an IC [7,16]. Laser is probably the most expensive fault injection means, however it makes it possible to inject faults with high accuracy even at advanced technology nodes [10]. It is accurate both in terms of timing (faults may be injected with laser pulses as short as a few picoseconds [12]) and in terms of location (it affects mainly the logic gate located within its spot size that may be as low as a few micrometers [7]). This explains why significant research work is dedicated to the study of laser-induced FAs.

The properties of the faults induced by laser (or by any other fault injection means) are referred to as a fault model (FM). It is often linked to a given attack scheme and expressed as an ability to meet requirements in terms of synchronization with the target activity and extension of the induced fault (e.g. the FM of the well-known Piret FA [14] requires to fault one byte of the AES algorithm calculations before its last MixColumn transformation).

In this work, we report our analysis of the laser-induced instruction skip FM. This FM relates to how a given instruction of a microcontroller program may be skipped (i.e. not executed) at runtime. Several works already described this FM and assessed the possibility of laser-induced single instruction skips [5,6,17]. The assessment of FMs on experimental grounds is of high interest regarding how FAs countermeasures (CMs) are designed and tailored. As a matter of example, the authors of [13] discussed two CMs based on instruction redundancy designed on the assumption that an attacker is only able to induce single instruction skips. Would this assumption be proved wrong, their CMs would be vulnerable.

Our experiments extend further this FM by reporting the feasibility of inducing several successive instruction skips by laser illumination. It also assesses the ability to skip several, but close, separate groups of instructions. This is a very strong FM which is very difficult to defend against with software only CMs.

This article is organized as follows. Section 2 discusses the state-of-the-art of instruction skips and introduces the aim of our work. Our experimental setup and settings are described in section 3. Section 4 reports the obtained results. Then, the assessed FM is discussed in section 5. Section 6 concludes the paper.

2 The laser-induced instruction skip fault model

2.1 Fault model definition

A FM generally describes the main properties of a FA scheme, often expressed in terms of requirements of synchronization (requirement to fault a particular step of an algorithm or program) and of extension (requirement to limit the fault extension, e.g. to a single bit or a single byte). In this work, we consider the FM related to laser fault injection. It may be defined at different levels of abstraction from transistor or gate level (as the authors of [10] did to describe laser-induced bit-set and bit-reset faults) to the assembler or algorithm level. We studied the FM of microcontrollers experiencing laser-induced instruction skips.

2.2 The instruction skip fault model, State-of-the-Art

An instruction skip is a fault that results in skipping, meaning not executing, one instruction of a microcontroller program at runtime (as if the program flow had skipped over the faulted instruction).

Several works studied the EM-induced instruction skip FM. Most of them assessed single instruction skips (in the same 8-bit microcontroller we used as target for [3], and on a 32-bit microcontroller for [13]). To the best of our knowledge, the only works reporting several successive instruction skips are [15] and [20]. [15] assessed four successive skips of instructions stored in the target instruction cache while [20] succeeded in faulting instructions stored in the target's pipeline.

Several works also deal with laser-induced instruction skips. [5, 6] obtained single instruction skips with high accuracy and high success rate (on the same microcontroller we studied) and used it to perform a successful differential fault

attack on AES. Still on the same target, [11] reports instruction skips based on resetting one or two bits of the targeted instruction opcode. The authors of [17] induced instruction skips on a more complex 32-bit cortex-M3 microcontroller. They were able to inject two single instruction skips distant from 58 ms to defeat a protected CRT-RSA algorithm. In terms of target complexity, [19] reports injection of single instruction skips into a quad core ARM cortex A9 microprocessor running at 1.4 GHz clock frequency. Hence, the state-of-the-art in laser-induced instruction skip was limited to single instruction skips (with a repetition rate in the range of tens of ms).

2.3 Study of laser-induced instruction skips

There is to date very few explanations of how an instruction skip is induced at the gate level, with the notable exception of [1]. It describes how increasing progressively the stress applied by a clock glitch to a microcontroller induces an increasing number of bit-reset faults into an instruction opcode. It results in (1) instruction modification at low stress or (2) in turning the instruction into an actual `nop` at high stress (i.e. the `no operation` instruction). Instruction modification achieves an instruction skip if the modified instruction has no effect on the code operations (instruction skips are often actual code modification); the same is true for turning an instruction into a `nop`. An analysis of single instruction skips due to instruction modification induced by laser is reported in [8]. It relates how faulting one bit of an instruction opcode led to two successful FAs.

Our research objective was to reproduce laser-induced instruction skips on a microcontroller and to study the main characteristics of its FM: accuracy, extent, success rate, time between successful skips, etc. Our aim was also to assess whether the single instruction skip fault model could be extended further to multiple instruction skips. This latter aim was of interest because CMs are based on the known FMs. Hence, a CM based on a too narrow FM may reveal vulnerabilities at test time. From previous experiments and taking into account the results of [1, 5, 6], we focused our experiments toward achieving instruction skips by turning the target instructions into `nop` instructions. Our experiments were carried out with the same 8-bit microcontroller studied by [1, 3, 5, 6, 11] for comparison purposes and also to ease the analysis of the obtained results.

3 Experimental settings

3.1 Laser bench

Laser source parameters Our laser source is a nanosecond range laser source able to output a laser pulse with a 50 ns to 1 s tunable duration. It has a latency of less than 300 ns (i.e. the time interval between the trigger signal and the moment an actual laser pulse hits the target). Its wavelength is 1,064 nm (or near infrared, NIR). The use of a NIR laser source allowed us to perform our laser fault injection experiments through the target's rear side because it is able to

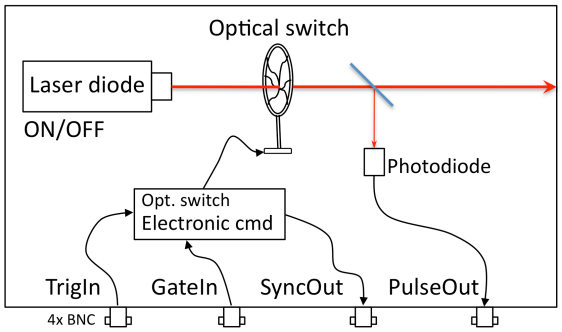


Fig. 1. Schematic of the main components of the used IR laser source.

pass through several hundreds of micrometers of silicon without being totally attenuated (the silicon die was also thinned down from 500 μm to 300 μm). The laser source max power is 3 W (measured at the fiber optic output) which is enough to inject faults into the target. We performed our experiments with a $\times 20$ objective lens: it outputs a laser spot diameter of 5 μm. An infrared camera was used to adjust the laser spot focus and location w.r.t. the target's layout.

Laser source technology Figure 1 provides a schematic view of the laser source technology. Any outputted laser pulse is carved from a continuous laser beam produced by a NIR laser diode thanks to an electro-optical switch (the laser diode operates on an ON/OFF basis). It takes 25 ns to open (resp. close) the optical switch of our source and to output a laser pulse that reaches the 3 W nominal power (resp. to extinguish a laser pulse). This explains that the shortest achievable laser pulse is 50 ns long. It also offers the ability to generate several consecutive laser pulses with a pause time in between as short as 50 ns. This repetition rate of 50 ns is a useful property for an attacker as underlined in subsection 4.4. An electronic board drives the optical switch (denoted Electronic cmd in Fig. 1). It features two laser shot modes, (1) the Trigger Mode and (2) the GateIn Mode:

- in Trigger Mode, a single laser pulse is produced in response to a voltage pulse delivered on the input BNC connector `TrigIn` (its duration, its power, and its delay w.r.t. the trigger signal are programmable),
- in GateIn Mode, the optical switch openings and closings follow the shape of the voltage signal applied on the input BNC connector `GateIn`. It makes it possible to generate a succession of voltage pulses of arbitrary shape (only constrained by the 25 ns open/close time of the switch).

A beam splitter captures a small amount of the laser pulse power that is converted into a voltage by a photodiode (see Fig. 1). It may be observed on the source `PulseOut` output BNC connector. This feature is useful to observe the synchronization of the actual laser pulse with the target activity.

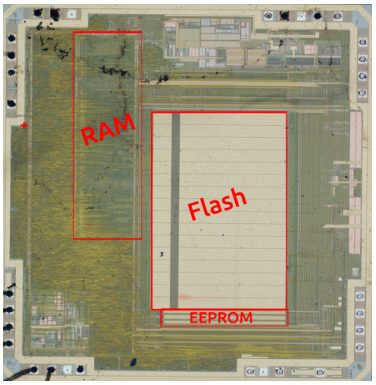


Fig. 2. Frontside view of the ATmega328P test chip - Flash, RAM and EEPROM memories highlighted in red.

Laser bench The target is placed on a XYZ stage (with 0.1 μm accuracy). An IR camera is used to observe the location of the laser spot. The activity of the target and the laser source signals are recorded with an oscilloscope.

Laser-Sensitivity Map Drawing Process Using a control PC to automate the process, we moved the laser over the area of the targeted device by displacement steps from 50 μm to 5 μm. For each position, we shot the laser while the microcontroller was running the test codes described in subsection 3.3. This allowed us to draw XY maps of laser-sensitivity: for each position where a fault was recorded, we drew a dot colored according to the obtained faults. Such laser-sensitivity maps were drawn for various laser power and timing.

3.2 Test chip

We chose a simple target for the purpose of being able to analyse easily its answers to fault injection: an 8-bit non-secure ATmega328P microcontroller designed in the old CMOS 0.35 μm technology. It has 2 kB RAM, 3 kB Flash and 1 kB EEPROM memories; a Harvard architecture with a 2-stage fetch-execute pipeline. It runs at 16 MHz and has 32 general purpose registers. Registers r16 to r25 were used during our experiments. Figure 2 gives a front view of the test chip with its Flash, RAM and EEPROM memories highlighted in red. A red cross near the device bonding pads on its left shows the XY origin we used as a reference.

3.3 Test codes

We studied the effect of laser-induced faults on dedicated test codes mostly written in assembly langage. Our intent was to induce and analyze instruction skips by examining their effect on the assembly instructions of the test codes.

For each test series, we used two trigger signals for synchronization purposes (two outputs of the test chip):

Listing 1 Test code - Instruction skip analysis.

```
# Store 0x39 to 0x30 in RAM at address Z
# Initialize r16 to r25 at 0x55
# Set synchronization trigger
nop # 400 ns
# Set core trigger
ld r16,Z+       ld r16,Z+
ld r17,Z+       ld r17,Z+
ld r18,Z+       ld r18,Z+
ld r19,Z+       nop
ld r20,Z+       ld r20,Z+
ld r21,Z+       ld r21,Z+
ld r22,Z+       ld r22,Z+
ld r23,Z+       ld r23,Z+
ld r24,Z+       ld r24,Z+
ld r25,Z+       ld r25,Z+
# Clear core trigger
nop # 700 ns
# Clear synchronization trigger
# read back r16 to r25
```

- a synchronization trigger signal to accommodate for the latency of the laser source,
- a core trigger signal to synchronize the actual laser shot (thanks to the `PulseOut` signal) with the part of the assembly code of interest.

Listing 1 provides a description of the test code we used to tune our settings in order to induce instruction skips. The core part of the test code (encompassed by the core trigger) is a series of ten `ld rX,Z+` instructions, each one corresponding to a load in a destination register `rX` of a byte value stored in RAM memory at address `Z` with a post increment of `Z`. Prior to that, the ten destination registers, `r16` to `r25`, are initialized at `0x55` and an array of ten byte values `0x39` to `0x30` are stored in RAM with `Z` storing the address of its first element. Registers `r16` to `r25` are read back after the synchronization trigger is reseted (the two top blue signals in Figure 4 are the synchronization and core triggers drawn for a fault free execution). The top part of Table 1 displays the values read back from `r16` to `r25` for a fault-free execution

Table 1. Registers r16 to r25 readback values, for a fault free execution (top) and for an instruction skip targeting r19 (bottom, highlighted in red).

Register	16	17	18	19	20	21	22	23	24	25
Fault free	0x39	0x38	0x37	0x36	0x35	0x34	0x33	0x32	0x31	0x30

Faulted	0x39	0x38	0x37	0x55	0x36	0x35	0x34	0x33	0x32	0x31

As an example, the right column of the test code core part in Listing 1 displays the effect of a laser shot turning the `ld` instruction of line 9 into a `nop` instruction. The effect of such a laser-induced instruction skip is highlighted in the bottom part of Table 1: the initialization value `0x55` is read back from `r19` (in red), and because an increment of address `Z` is missing, all the values read back from `r20` to `r25` are shifted (in gray).

4 Experimental results

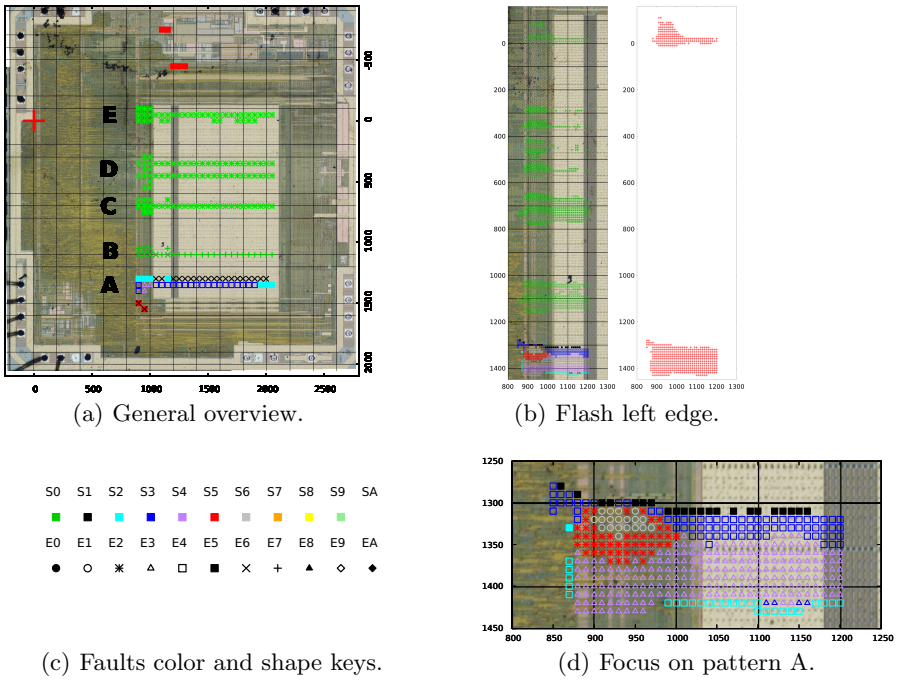
4.1 Finding the Points-of-Interest

General overview Our first series of experiments aimed at finding a point where laser illumination of our target would induce an instruction skip in the test code of Listing 1. Using the synchronization trigger, the delay was set to target the `ld` operation into register `r19`. The laser pulse duration was set to 200 ns (a little more than three clock periods) and its power to 0.5 W. In order to gain a first insight of potential points of interest (i.e. inducing instruction skips), we scanned the whole target area with XY steps of 50 μm. The obtained results are depicted in Fig. 3(a), while the color and shape code describing the faulty behavior is given in Fig. 3(c). The color denotes the number of registers storing the initialization value at read back indicating that the corresponding `ld` operations were not executed (from no register noted S0 to 10 registers noted SA), the shape indicates the number of other incorrect values stored into the registers at read back (from 0 noted E0 to 10 noted EA). As an example, the instruction skip exemplified in the bottom part of table 1 is depicted by a black $\times$ shape (`r19` storing the initialization value and `r20` to `r25` storing six incorrect values, i.e. S1E6).

We identified the following faulty behaviors:

- Red squares on top near an analog block, locations for which the registers read back was all zero. We did not study further this faulty mechanism.
- Red barred squares near the bottom left corner of the Flash memory for which communication with the test chip was lost (until reset).
- Horizontal patterns spanning along the width of the Flash memory (identified with letters from A to E) which were consistent with instruction skips as explained in table 1.

Spatial focus on points of interest Our next experiments focused on the left edge of the Flash memory with an increased spatial accuracy: XY step set to 10 μm. On its left, figure 3(b) reports the observed fault model for a register initialization value of 0x55. The green color of patterns B to E revealed fault models that are not consistent with an instruction skip (the initialization value was not found in any of registers r16 to r25 at read back). We do not report any further analysis of the corresponding fault models since this research work focuses on instruction skips. Note that the authors of [11] obtained similar fault patterns on the same microcontroller, which they identified as bit reset faults induced at read back on the 16-bit opcode instructions.



(a) General overview. (b) Flash left edge.

(c) Faults color and shape keys. (d) Focus on pattern A.

Fig. 3. Laser sensitivity maps obtained while running `Listing 1` test code: general overview (a) and focused overviews (b, d); fault key (c).

Instruction skip sensitive area Fault pattern A reveals a different behavior (see sensitivity map of Fig. 3(d)). Several consecutive instruction skips, from one (depicted in black) to six (depicted in grey), were obtained. The corresponding shapes also revealed that the skips where followed by shifts of the values stored into the registers following the skipped registers. In addition, the right map in figure 3(b) shows with red crosses the fault locations where the duration of the trigger signals was shortened by one or more clock periods. It further reinforces our analysis that the instruction skips in pattern A are obtained by turning the `ld` instructions into `nop` instructions. Because execution of a `ld` instruction takes two clock periods contrary to a `nop` instruction which takes one clock period, each consecutive instruction skip shall correspond to a reduction of the test code of one clock period. This phenomenon is displayed in figure 4 for a single instruction skip. The test code execution time is shortened as well as the duration of the triggers signals: the fault free execution triggers in blue last one clock period more than the faulted execution that is drawn in red (the third signal, in red, is the laser source `PulseOut` output which shows the actual timing of the laser pulse). For each fault injection location of pattern A, we observed a shortening of the trigger signals equals to the number of instruction skips multiplied by the clock period.

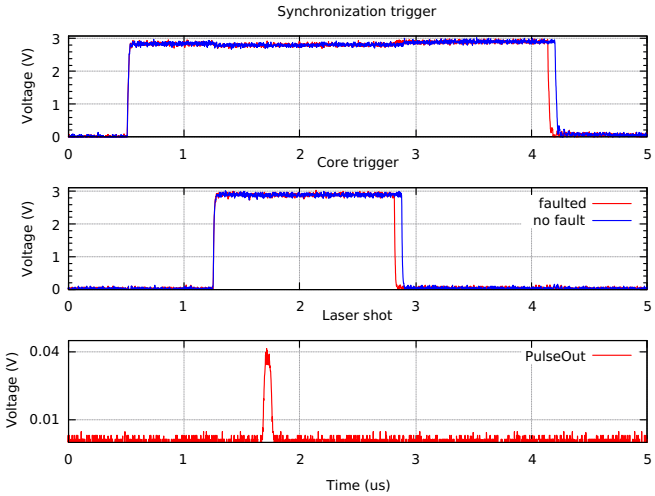
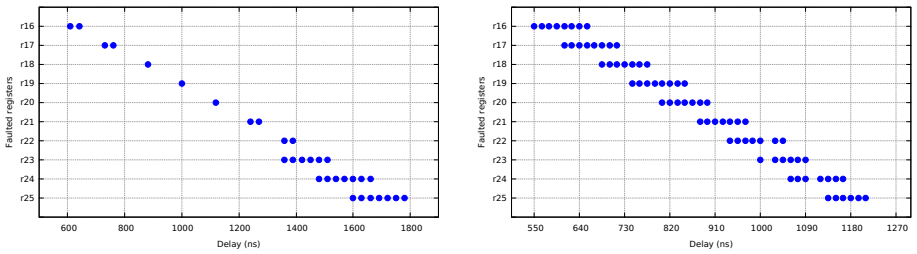


Fig. 4. Synchronisation trigger signals with and without a laser shot (depicted in red and blue resp.), and PulseOut laser source output signal.

4.2 Laser-induced instruction skip, test of accuracy

The left part of pattern A, lying outside the Flash memory, seemed more promising in terms of ability to induce instruction skips. We performed there a set of experiments with an increased XY step accuracy of 5 μm and laser settings ranging from 0.2 W to 0.5 W (0.1 W step) and 50 ns to 125 ns duration (25 ns step). Our objective was to find parameters allowing to induce single instruction skips with high repeatability. We were indeed able to find several locations where we induced such single instruction skips on `r19` with a 100 % success rate (the laser parameters were set to 75 ns pulse duration and 0.4 W power). These locations, close to (950 μm ; 1,350 μm) (see map of Fig. 3(d)), were used for the experiments reported hereafter.

In terms of accuracy, we also tested whether this single instruction skip fault model was still valid while targeting the `ld` instruction of the other test registers. Our aim was to assess an attacker ability to target arbitrarily a single instruction of a program. To do so, we varied the time delay between the laser shot and the synchronization trigger signal to span the whole test code. Figure 5(a) reports the obtained results. It displays the skipped registers as a function of the delay. It reveals that an attacker is able to inject laser-induced single instruction skips into a running microcontroller with high timing accuracy. For each instruction, we were able to find an injection timing leading to 100 % success rate (this success rate may be lower than 100 % while passing from one instruction to the other). At some timings, two consecutive instructions were skipped (e.g. at 1,390 ns or 1,625 ns in Fig. 5(a)), suggesting that several consecutive instructions may be skipped simultaneously. For the purpose of verifying this suggestion, we again carried out our experiments with a laser duration increased to 125 ns (two clock



(a) (b)

Fig. 5. Faulted registers as a function of laser injection time: laser power of 0.4 W and duration of 75 ns (a) and 125 ns (b).

cycles) and a delay step set to 20 ns. The corresponding results are displayed in figure 5(b). It shows that increasing the laser duration to 125 ns makes it possible to skip two consecutive instructions with a still high timing accuracy (i.e. the ability to choose the two skipped instructions).

4.3 Laser-induced arbitrary number of instruction skips

We also tested whether increasing the laser pulse duration would make it possible to skip an arbitrary number of consecutive instructions. The laser power was kept constant at 0.4 W, and the delay was set to target the `ld` instruction of register `r19`. The test series were carried out for a pulse duration ranging from 50 ns to 410 ns with an increment step of 30 ns. Figure 6 reports the obtained results.

A first instruction skip was obtained for a laser pulse duration of 80 ns. Then, the number of instruction skips increased progressively with the pulse duration up to 7 consecutive skips at 350 ns. On average an additional instruction skip was obtained for every 60 ns increment of the laser pulse duration. For each number of instruction skips between 1 and 7, we were able to find settings leading to a 100 % success rate.

This suggested that a fault model for which an attacker has the ability to skip an arbitrary number of code instructions (i.e. a chosen number) is feasible. We used the test code shown in Listing 2 to ascertain the highest possible number of consecutive skips. Its structure is similar to the code of listing 1: a set of target assembly instructions marked by a core trigger encompassed by a larger synchronization trigger with `nop` instructions in between. We chose the `adiw`, or add immediate to word, instruction as target. The target part of the test code had several successive `adiw` instructions used to increment by one the 16-bit word stored in the `r25:r24` registers pair. We chose it over other addition instructions because it lasts two clock cycles, hence each `adiw` faulted into a `nop` shall be ascertained both by the final value stored in `r25:r24` and by a shortening of the trigger signals.

For our experiments, `r25:r24` was initialized at 0x0000 and the sequence of `adiw` instructions set to 300. The laser power and delay were set to 0.5 W and 800 ns. As we increased progressively the laser pulse duration, an increased

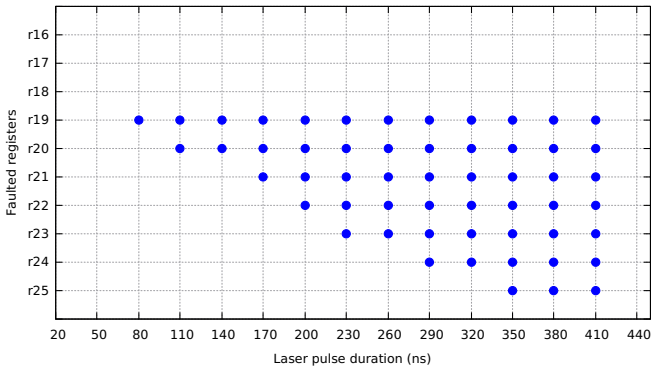


Fig. 6. Faulted registers as a function of the laser pulse duration (laser duration from 50 ns to 410 ns, 0.4 W laser power).

Listing 2 Test code - Max. number of instruction skips.

```
# Initialize r25:r24 at 0x0000
# Set synchronization trigger
nop # 300 ns
# Set core trigger
adiw r24,0x01
...  # 300 times
adiw r24,0x01
# Clear core trigger
nop # 300 ns
# Clear synchronization trigger
# read back r25:r24
```

number of `adiw` instructions were skipped, for which the shortening of the trigger signals was in accordance with the number of missing additions into `r25:r24`. Table 2 gives the number of obtained successive instruction skips for a selection of laser pulse durations. It took a 20,400 ns long laser pulse to skip the whole 300 `adiw` instructions. Note that the reproducibility of the experiments decreased a bit as the laser pulse duration increased: the number of induced skips varied from 1 or 2 skips at 1,000 ns to 4-5 skips above 10,000 ns from one experiment to the other. We did not test the number of skipped instructions beyond 300. There is a maximal number of instruction skips set by the endurance to laser illumination of the target circuit. Indeed, our device was destroyed when accidentally exposed

Table 2. Number of obtained instruction skips vs laser pulse duration

Laser pulse duration (ns)	1,000	2,000	5,000	10,000	20,400
Number of instr. skips	17	33	82	143	300

to a continuous laser pulse at the same 0.5 W power. However, the device we used for these experiments showed no sign of fatigue after several tests at 20,400 ns laser pulse duration.

4.4 PIN bypass with several laser pulses

The technology of our laser source (see description given in subsection 3.1) makes it possible to carve several consecutive pulses in the continuous laser beam delivered by a laser diode. Using such a sequence of laser pulses an attacker might be able to skip several sections of arbitrary length in the target's firmware.

In order to assess the feasibility of this fault injection technique, we targeted a 4-digit PIN verification algorithm (described in [9]). It is protected against side channel timing analysis by a constant-time implementation: every of the digits entered by the user are compared with those of a reference PIN. The four corresponding comparison loops are shown in Listing 3, where `i` is the index of the user and reference PIN arrays (resp. `a1[i]` and `a2[i]`), `PINSIZE` the PIN code length, `BOOL_TRUE` and `BOOL_FALSE` are resp. the true and false boolean values, and `diff` a variable indicating whether the user and reference PINs differ or not. `diff` set to `BOOL_TRUE` indicates that the user and reference PINs are different: as a result the user identification will be rejected (this part of the code is not shown). `diff` is initialized at `BOOL_FALSE` before running the PIN arrays compare loops.

Listing 3 C code of the PIN arrays compare loops.

```
1  BOOL diff = BOOL_FALSE;
2  ...
3  for(i = 0; i < PINSIZE; i++) {
4   if(a1[i] != a2[i]) {
5    diff = BOOL_TRUE;
6   }
7  }
```

Laser-induced fault injection may be used to force the identification of an attacker using a wrong PIN. As illustrated in [8], where a laser-induced modification of one instruction of a similar PIN algorithm permits to perform a PIN bypass. [9] describes, on simulation basis, several instruction skip attacks that may result in a successful PIN bypass: most of them targeting one or a few successive code instructions. We chose to implement that which consists in skipping the four instructions in charge of setting `diff` to `BOOL_TRUE` when a false user PIN is used (line 5 of Listing 3). This fault model is often considered as unlikely because the laser pulse repetition rate of laser sources may be too long. However, our laser is able to meet the 875 ns duration of the PIN compare loops (it is able to emit successive pulses in less than 50 ns). We were able to synchronize four 60 ns long laser pulses in order to skip the instructions used to

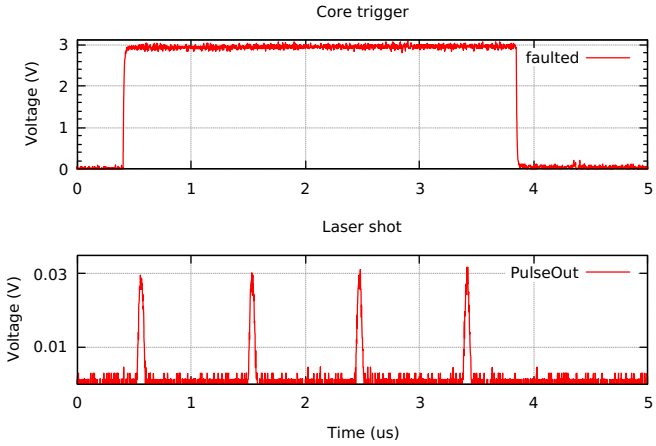


Fig. 7. Laser-induced PIN code bypass induced by four laser pulses emitted in a row (4×60 ns, 0.5 W, 875 ns interval).

set `diff` to `BOOL_TRUE` (with a 0.5 W laser power) and gain identification with a false user PIN. Figure 7 displays the core trigger signal encompassing tightly the comparison loops and the `PulseOut` signal showing the actual timing and shape of the four laser pulses.

5 Discussion

5.1 Laser-induced instruction skip fault model

A powerful fault model This work, based on experimental results, studied the instruction skip FM obtained on an 8-bit microcontroller exposed to laser illumination. It demonstrates that a very high accuracy is achievable: we were able to choose and skip a single instruction into a test sequence with a 100 % success rate. Moreover, what gives a particular strength to this FM (and where its novelty lie w.r.t. [15,20]) is the ability, by increasing the duration of the laser pulse, to skip an arbitrary number of consecutive instructions of the target's firmware. Provided an attacker is able to find out the adequate laser settings, he shall be able to erase an entire section of a program. Using a laser source with low repetition rate (less than 50 ns in our case), we were also able to skip instructions in the four comparison loops of a PIN code verification algorithm. It made it possible to bypass the PIN verification. This further extends the FM's ability to skip (i.e. erase) several sections of arbitrary length of a microcontroller firmware at runtime.

The strength of this fault model may questions the feasibility of software CMs (a question already raised by [20]): if an attacker has the ability to erase arbitrary sections of a firmware, he will also skip the software CMs it contains. However, secure ICs embed various types of CMs (e.g. laser sensors, hardware redundancy, etc.) that are not put at risk by this FM.

Denomination of the fault model The laser-induced instruction skip FM relies on two possible mechanisms: (1) injecting faults into an instruction opcode [8,11] that turn it into another instruction, or (2) replacing it by a `nop` instruction (this work). In neither case is the instruction really skipped over. The faults we obtained may rather be described as `nop`-ization or as instruction erasure. However, we stuck with the instruction skip denomination for the purpose of avoiding confusion; though, the underlying phenomenon is not an actual skip.

5.2 Synchronization

The fault model we explored is powerful both in terms of accuracy (the ability to skip a single chosen instruction) and of extent (ability to skip several consecutive instructions). However, we used a white box approach for which we used trigger signals to synchronize the laser shots with the test codes. The lack of a trigger signal shall be one of the main difficulty to be tackle with for a real attack case: difficulty of synchronization constitutes a useful and effective counter-measure.

Though, the use of a smart trigger based on real-time pattern recognition of a side channel signal (e.g. the power consumption of the target) may allow an attacker to obtain an accurate synchronization with the operations of its target (as in [18]). Then, the use of a first synchronization may be used to perform a complex attack. As an example, consider the PIN bypass case described in subsection 4.4. An attacker may reverse a PIN verification algorithm in a black box scenario for which the PIN algorithm is unknown, provided he owns a valid identification PIN in order to reinitialize periodically the number of PIN trials (it is usually limited to three). This is a four steps process (considering a four digits PIN code). The first step consists in synchronizing a first laser pulse with the first compare loop (see Listing 3) by using a PIN code having a first false digit and three other correct digits. The first synchronization step is achieved when the PIN identification is obtained. Then, iteratively the same process may be used one digit after the other to obtained a full synchronization of the attack. As soon as, the time profile (and laser settings) of a successful PIN bypass is known to the attacker, he will be able to reproduce the attack and gain illegitimate access to whatever is protected by the same PIN verification algorithm.

5.3 Generalization

Our experiments were carried out on a non-secure AVR 8-bit microcontroller which architecture was introduced in 1997. This raises the question of the generalization of the obtained FM to up-to-date microcontrollers. A firm answer to this question is certainly to be based on actual experiments on other targets (e.g. 32-bit microcontrollers).

However, several research works provide indications in favor of generalization. [17] reports laser-induced single instruction skips obtained on a 32-bit cortex-M3 target. The laser sensitive area they identified is located close to the Flash memory of their target, in a similar way of our work. This suggest that a similar mechanism may be at work, even though they did not test the feasibility of

successive instruction skips. The authors of [8] also induced single instruction skips through laser illumination into a cortex-M3 microcontroller by faulting a single bit of the targeted instruction (hence inducing an instruction modification equivalent to an instruction skip because the modified instruction had no effect on the executed test code). The physical mechanism they revealed as the root cause of this instruction skip (a laser-induced discharge of a bitline of the Flash memory) appears compatible with induction of several skips if a long laser pulse is used. In terms of accuracy, [10] reports that single bit faults may be induced by laser in targets designed in a technology as advanced as the 28 nm CMOS process. These different research works suggest that a FM for which a single or several successive instruction skips may be feasible with careful tuning of a laser shot parameters and should be considered when designing a secure circuit.

6 Conclusion

This research work assesses on experimental basis an extended fault model for laser-induced instruction skips. The main characteristics of this fault model are:

- its accuracy, or ability to choose the skipped instruction with a 100 % success rate provided a precise synchronization is obtained,
- its extension, or ability to skip an arbitrary number of successive instructions,
- its flexibility, or ability to skip several sections of the targeted firmware.

Simply put, laser FA may offer an attacker the ability to erase chosen parts of a microcontroller firmware at runtime. Generalization of this FM beyond the case of our 8-bit target is to be proven, though several results provides arguments in favor of such a possibility. However, this first experimental assessment speaks in favor of considering it when designing CMs. A task that may prove difficult to complete given the assumption that any part of a software CM might be skipped.

Acknowledgment

This research has been partially supported by the European Commission under H2020 SPARTA (Grant Agreement 830892)

References

1. Balasch, J., Gierlichs, B., Verbauwhede, I.: An in-depth and black-box characterization of the effects of clock glitches on 8-bit MCUs. In: Fault Diagnosis and Tolerance in Cryptography (2011)
2. Barenghi, A., Breveglieri, L., Koren, I., Naccache, D.: Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures. Proceedings of the IEEE **100**, 3056 – 3076 (2012)
3. Beckers, A., Balasch, J., Gierlichs, B., Verbauwhede, I., Osuka, S., Kinugawa, M., Fujimoto, D., Hayashi, Y.: Characterization of EM faults on ATmega328P. In: International Symposium on Electromagnetic Compatibility. IEEE (2019)

4. Boneh, D., DeMillo, R.A., Lipton, R.J.: On the importance of checking cryptographic protocols for faults. In: Advances in Cryptology, International Conference on the Theory and Application of Cryptographic Techniques (1997)
5. Breier, J., Jap, D.: Testing feasibility of back-side laser fault injection on a microcontroller. In: Proceedings of the WESS'15: Workshop on Embedded Systems Security. New York, NY, USA (2015)
6. Breier, J., Jap, D., Chen, C.N.: 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. New York, NY, USA (2015)
7. Buchner, S., Miller, F., Pouget, V., McMorrow, D.: Pulsed-laser testing for single-event effects investigations. Nuclear Science, IEEE Transactions on **60**(3), 1852–1875 (June 2013)
8. Colombier, B., Menu, A., Dutertre, J.M., Moëllic, P.A., Rigaud, J.B., Danger, J.L.: Laser-induced single-bit faults in flash memory: Instructions corruption on a 32-bit microcontroller. In: Hardware-Oriented Security and Trust (2019)
9. Dureuil, L., Petiot, G., Potet, M.L., Le, T.H., Crohen, A., de Choudens, P.: FISSC: A fault injection and simulation secure collection. In: International Conference on Computer Safety, Reliability, and Security (2016)
10. Dutertre, J.M., Beroulle, V., Candelier, P., De Castro, S., Faber, L.B., Flottes, M.L., Gendrier, P., Hély, D., Leveugle, R., Maistri, P., Di Natale, G., Papadimitriou, A., Rouzeyre, B.: Laser fault injection at the CMOS 28 nm technology node: an analysis of the fault model,. In: 2018 Workshop on Fault Diagnosis and Tolerance in Cryptography (2018)
11. Kumar, D., Beckers, A., Balasch, J., Gierlichs, B., Verbauwhede, I.: An in-depth and black-box characterization of the effects of laser pulses on ATmega328P. In: Smart Card Research and Advanced Applications (2018)
12. Lacruche, M., Borrel, N., Champeix, C., Roscian, C., Sarafianos, A., Rigaud, J.B., Dutertre, J.M., Kussener, E.: Laser fault injection into SRAM cells: Picosecond versus nanosecond pulses. In: On-Line Testing Symposium (2015)
13. Moro, N., Heydemann, K., Dehbaoui, A., Robisson, B., Encrenaz, E.: Experimental evaluation of two software countermeasures against fault attacks. In: Hardware-Oriented Security and Trust (2014)
14. Piret, G., Quisquater, J.J.: A differential fault attack technique against SPN structures, with application to the AES and Khazad. In: Cryptographic Hardware and Embedded Systems. Berlin, Heidelberg (2003)
15. Rivière, L., Najm, Z., Rauzy, P., Danger, J.L., Bringer, J.: High precision fault attacks on the instruction cache of ARMv7-M architectures. In: 2015 IEEE International Symposium on Hardware Oriented Security and Trust (2015)
16. Skorobogatov, S.P., Anderson, R.J.: Optical fault induction attacks. In: 4th International Workshop on Cryptographic Hardware and Embedded Systems (2002)
17. Trichina, E., Korkikyan, R.: Multi fault laser attacks on protected CRT-RSA. In: Fault Diagnosis and Tolerance in Cryptography (2010)
18. van Woudenberg, J.G.J., Witteman, M.F., Menarini, F.: Practical optical fault injection on secure microcontrollers. In: 2011 Workshop on Fault Diagnosis and Tolerance in Cryptography (2011)
19. Vasselle, A., Thiebeauld, H., Maouhoub, Q., Morisset, A., Ermeneux, S.: Laser-induced fault injection on smartphone bypassing the secure boot. In: 2017 Workshop on Fault Diagnosis and Tolerance in Cryptography (2017)
20. Yuce, B., Ghalaty, N.F., Santapuri, H., Deshpande, C., Patrick, C., Schaumont, P.: Software fault resistance is futile: Effective single-glitch attacks. In: 2016 Workshop on Fault Diagnosis and Tolerance in Cryptography (2016)