PAPER *Special Section on Cryptography and Information Security*

Laser-Induced Controllable Instruction Replacement Fault Attack

Junichi SAKAMOTO[†a)], ***Student Member***, **Daisuke FUJIMOTO**[†b)], ***and*** **Tsutomu MATSUMOTO**[†c)], ***Members***

SUMMARY To develop countermeasures against fault attacks, it is important to model an attacker's ability. The instruction skip model is a well-studied practical model for fault attacks on software. Contrastingly, few studies have investigated the instruction replacement model, which is a generalization of the instruction skip model, because replacing an instruction with a desired one is considered difficult. Some previous studies have reported successful instruction replacements; however, those studies concluded that such instruction replacements are not practical attacks because the outcomes of the replacements are uncontrollable. This paper proposes the concept of a controllable instruction replacement technique that uses the laser irradiation of flash memory. The feasibility of the proposed technique is demonstrated experimentally using a smartcard-type ARM SC100 microcontroller. Then, practical cryptosystem attacks that exploit the proposed technique are investigated. The targeted cryptosystems employ the AES with software-based anti-fault countermeasures. We demonstrate that an existing anti-instruction-skip countermeasure can be circumvented by replacing a critical instruction, e.g., a branch instruction to detect fault occurrence.

key words: *instruction replacement, instruction skip, fault attack, laser fault injection, side-channel attack*

1. Introduction

The phrase "Trillion Sensor Universe," which is increasingly used in the literature, suggests that the Internet of Things era will involve an extremely large number of devices. In this situation, physical attacks on cryptosystems will increase due to relatively easy physical access to devices. Physical attacks on cryptosystems can be classified as; side-channel attacks (passive attacks), and fault attacks (active attacks). This paper focuses on fault attacks, which obtain stored secret data by intentionally injecting "faults" into the electronic circuit. For example, such faults can include forcible output of secret data. To develop countermeasures against fault attacks, it is necessary to restrict attacker's ability. This process is referred to as "attacker modeling," and the modeled results are referred to as a "fault model."

In this study, we focus on instruction skip and instruction replacement models. The instruction skip model faults, which allows an attacker to skip an assembly language instruction, has been observed on several embedded processor architectures and for several fault injection methods [1]–[5]. Actually, the instruction skip model, a widely studied common fault model, is considered a subset of the instruction replacement model, in which attackers can replace an assembly language instruction. Here, replacements to No Operation (NOP) instructions correspond to the instruction skip [6]. Previous studies have reported that instruction replacement faults as well as some instruction skip faults occur on different architectures and through different fault injection means [1], [3]. However, these faults should be referred to as instruction "corruption" rather than "replacement" because replacing an instruction with one the attackers want is difficult.

We propose the concept of a controllable instruction replacement technique where a running instruction can be change to a desired instruction. We demonstrate the feasibility of the proposed concept experimentally on a smartcard-type ARM SC100 microcontroller. The primary contributions of this study can be summarized as follows.

(I) Concept of controllable instruction replacement. To the best of our knowledge, the concept of how to inject controllable instruction replacement faults has not been described previously. In the proposed conceptualization, faults are injected into instruction codes being fetched using laser irradiation on the flash memory from which the target instruction codes are fetched. By using a laser to irradiate a sense amplifier on flash memory, it is expected that a logic value (1/0) issued by the sense amplifier will become faulty. Here, we assume that the *n*th sense amplifier reads the *n*th bit of instruction code; thus, aiming the laser at sense amplifiers enables us to control the position of the faulty bit in a replaced instruction.

(II) Demonstration of the feasibility of the proposed controllable instruction replacement concept. Laser experiments to demonstrate the concept of controllable instruction replacement were performed on an ARM secure processor. A laser irradiation environment was set up to control four laser parameters, i.e., irradiated point, duration, power and timing, to investigate how those parameters affect instruction replacement.

(III) Circumventing existing countermeasures against instruction skip using the controllable instruction replacement. Most previous studies on fault attacks against software consider instruction skip as a general fault model and propose attacks using and countermeasures against instruction skip. Instruction replacement model is an upper level notion of instruction skip model. Thus, the proposed controllable instruction replacement could com-

Manuscript received March 18, 2019.
Manuscript revised July 4, 2019.

[†]The authors are with Yokohama National University, Yokohama-shi, 240-8501 Japan.
a) E-mail: sakamoto-junichi-mb@ynu.jp
b) E-mail: fujimoto-daisuke-ht@ynu.ac.jp
c) E-mail: tsutomu@ynu.ac.jp

DOI: 10.1587/transfun.2019CIP0028

promise the security of existing countermeasures against instruction skip attacks. We describe instruction replacement attacks that circumvent existing countermeasures against instruction skip.

The remainder of this paper is organized as follows. Section 2 provides a brief overview of previous studies related to fault attacks and fault model. Section 3 introduces the concept of controllable instruction replacement based on laser irradiation of a sense amplifier in flash memory (the contribution (I)). The experimental setup used to demonstrate the feasibility of the proposed concept is described in Sect. 4. In addition, experimental results using an ARM SC100 processor are provided (the contribution (II)). In Sect. 5, we describe an attack on existing countermeasures against instruction skip (the contribution (III)). Conclusions and suggestions for future work are given in Sect. 6.

2. Related Works

To construct countermeasures against fault attacks, an attacker's ability is abstracted as a "fault model." As shown in Table 1, fault models are classified according to the level at which the faults occur. As mentioned previously, this study focuses on instruction skip and instruction replacement models.

2.1 Instruction Skip Model

Previous studies have reported the occurrence of instruction skip faults for several fault injection means on various devices, such as the clock glitch on the 8-bit AVR ATMega163 [1] and LEON3 processor [2], electromagnetic irradiation on the 32-bit ARM Cortex-M3 [3] and the 32-bit ARMv7-M device [4], and laser irradiation on the 32-bit ARM Cortex M3 [5]. Moreover, instruction skip faults cause upper-level IF skip and Loop Count skip faults. Differential fault analysis (DFA) techniques, which computes the candidates of a secret key with sets of a correct (fault-free) ciphertext and an incorrect (fault-injected) ciphertext, are often proposed based on these algorithmic level fault models; thus, the instruction skip model is considered to be a practical and common threat to software-implemented cryptosystems.

2.2 Instruction Replacement Model

In the instruction replacement model, attackers can replace a given instruction with a different instruction. The instruction skip model can be considered part of the instruction replacement model wherein a target instruction is replaced with a NOP instruction. Compared to the instruction skip model, the instruction replacement model represents significant threat. As mentioned previously, some experimental results reported that instruction replacement faults were observed on some devices. However, few studies focus primarily on instruction replacement and none consider developing countermeasures against the instruction replacements. Balasch et al. reported the occurrence of instruction replacements on an ATMega163 smartcard via a clock glitch at the time of instruction pre-fetch [1]. In their study, some instructions, e.g., EOR (Exclusive OR) and SER (SEt all bits in Register), were targeted, and the instructions after replacement depend on the period of the clock glitch. However, Balasch et al. concluded that controlling instructions after replacement is highly complex because it depends on the internal state of the MCU. Similar results were observed by Moro et al. on an ARM Cortex-M3 microcontroller through electromagnetic glitches [3]. Similar to Balasch et al., Moro et al. could not control instructions after replacement. On an ARM9 microcontroller, Barenghi et al. achieved more practical instruction replacements through underpowering the target device [7]. In their experiments, 1-bit stuck-at-0 faults were observed during instruction fetch, and they claimed that their method could produce instruction replacements, such as replacing an AND instruction with an EOR instruction. However, as their proposed method reduces the supply voltage during entire program execution, it is not possible to select instructions to be fault-injected (i.e. instruction replacement timing).

Table 1 Fault models. The lower (the bottom in the figure) level faults causes the upper level faults.

Fault level	Fault model	Main target
Algorithmic	IF skip	Software
	Loop Count skip	
Instruction	Instruction skip	
	Instruction replacement	
Register	Bits stuck-at/flip/random	Hardware
Transistor	Set/reset/flip	

Table 2 Comparison of the proposed method and previous instruction replacement studies.

	Fault injection mean	Instruction replacement timing	Instructions after replacement
[1]	Clock glitch	Controllable	Not controllable
[3]	EM irradiation	Controllable	Not controllable
[7]	Underpowering	Not controllable	Controllable
Ours	Laser irradiation	Controllable	Controllable

Table 2 compares the proposed method to previous studies about instruction replacement. As can be seen, instruction replacements in the previous studies can only control either instructions replacement timing or instructions after replacement. We propose a controllable instruction replacement method that can control both instruction replacement timing and instructions after replacement.

3. Proposed Method: Controllable Instruction Replacement by Irradiating Sense Amplifier in Flash Memory with Laser

This section explains the basic concept behind producing the controllable instruction replacement, i.e., the laser irradiation of a sense amplifier on a flash memory chip. Most electronic devices with cryptographic modules include flash

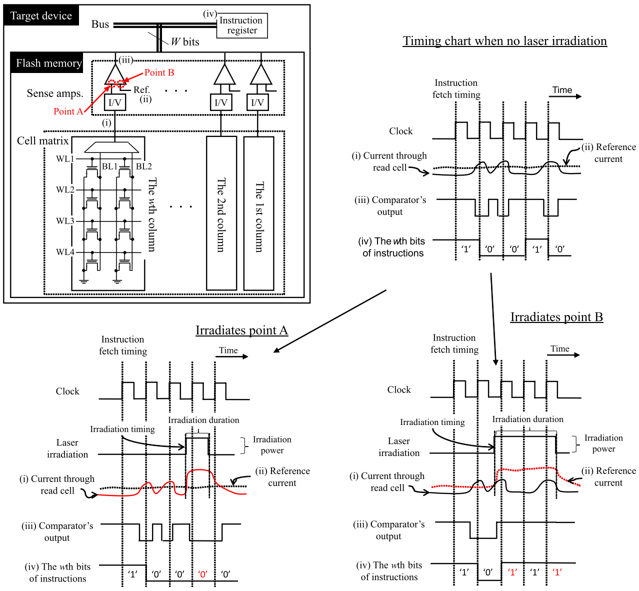


Fig. 1 Abstract structure of flash memory with w-bit data bus and timing chart of the sense amplifier operation under laser irradiation. The cell matrix is separated into w columns with each column connected to a sense amplifier. Laser irradiation on the wth sense amplifier produces a fault in the wth bit of the binary instruction code.

memory to store binary programs. However, it is easy to imagine that laser irradiation of flash memory can corrupt the instructions read from the memory.

Prior to introducing the proposed method, we first explain the structure and read operation of a typical flash memory device. Note that the proposed method targets NOR-type flash memory and a non instruction cache architecture. Figure 1 shows flash memory implemented on a target device with a w-bit data bus (one instruction = w bits). The flash memory has w columns in the cell array, and the cells in the nth column contain the nth bits of the binary instructions. The flash memory read operation begins with selecting a memory cell that corresponds to the read address of each column, by asserting a word line (WL) and pre-charging a bit line (BL). When the selected cell stores “0,” the gate of the cell is open. Therefore, the pre-charged electrons are drawn to GND through the cell. When the selected cell stores “1,” the gate of the cell is closed; thus, the pre-charged electrons remain on the bit line. As the current flow on the bit line is too weak to be handled as a single end signal, the sense amplifier amplifies the signal. A typical sense amplifier comprises two circuits: a current-to-voltage (I/V) converter and a comparator. The comparator compares the voltage produced by the I/V converter to a reference voltage value. Here, the comparator issues “1” if the reference voltage is the greater of the two values; otherwise, the comparator issues “0”. The comparator is implemented as a differential amplifier, which amplifies the weak input differential signal by approximately 100 times and issues the logic value. The comparator also likely amplifies the current generated by the laser irradiation up to approximately 100 times. Therefore, we expect to achieve fault injection by irradiating sense amplifiers with a relatively weak (and inexpensive) laser.

In the following, we describe how the proposed method (laser irradiation of sense amplifier in flash memory) injects instruction replacement faults and controls them. Figure 1 (top right) shows an example timing chart for the sense amplifier operation without laser irradiation. Here, the bit sequence “10010” is read from the wth column. This sequence corresponds to the wth bits of instruction codes to be executed. Consider an example in which point A, which is the input port of the wth comparator (Fig. 1) is irradiated with a laser. The timing chart is shown in Fig. 1 (bottom left). The laser irradiation begins just before the rising edge of the fourth clock and finishes before the rising edge of the fifth clock. Here, electron and hole pairs generated by the laser increase the current flow through the read cell during irradiation. Then, the comparator issues erroneous values because the input current is greater than the reference current. If the period of the comparator’s erroneous output spans the duration of an instruction fetch timing, a faulty value (“0” in this case) is latched in the instruction register. Since the latched value corresponds to the wth bit of the instruction to be executed, the irradiation of point A injects an instruction replacement fault, which makes the wth bit of the instruction “0.” In addition, the irradiation timing just before the rising edge of the fourth clock injects a fault into the instruction fetched at the fourth clock because the laser duration spans the instruction fetch timing. From this result, we predict that laser irradiation timing can control which instruction is replaced.

Next, consider laser irradiation of point B (the reference input of the wth comparator). The timing chart is shown in Fig. 1 (bottom right). The laser irradiation begins just before the rising edge of the third clock, and finishes after the rising edge of the fifth clock; thus, the laser duration crosses the instruction fetch timing three times. Here, the laser irradiation increases the reference current. As a result, the comparator’s output is fixed High value during the irradiation, and the faulty sequential value ‘110111” is latched. Although laser duration spans three instructions fetch timings, only 2 bits are replaced with “1”. This indicates that the faults injected by the proposed method are stuck-at faults. In addition, as the laser irradiation duration increases, the number of affected instruction fetch timings increases; therefore, we predict that laser irradiation duration controls the number of instruction replacements. Moreover, although a stuck-at-0 fault occurs at point A, a stuck-at-1 fault occurs at point B, which indicates that the irradiated point in the sense amplifier controls which faults are injected. Laser irradiation of the nth sense amplifier injects stuck-at faults in the nth bit of instruction codes; thus, the irradiated point can control which bit of the instruction code is injected with a fault. If multiple points are irradiated with laser, multiple bits of the instruction codes can be replaced. This fact provides important insight relating to an attacker’s ability. For example, consider an attacker seeking to replace instruction Ins_s with instruction Ins_d. The number N of irradiated point required to replace Ins_s with Ins_d is the Hamming distance between the binary representation of Ins_s and Ins_d. A greater N value enhances the attacker's ability but increases the cost of the attack. For commercially available laser attack evaluation setup, $N = 1$ or $N = 2$. If N is greater than two, the optics becomes too complicated to design and construct a feasible experiment. In the experiments discussed in Sect. 4, we demonstrate the proposed method in the case of $N = 1$.

In the proposed controllable instruction replacement method, the following laser parameters control instruction replacement: (i) the laser irradiation point controls which bits are replaced with which value ("0" or "1"); (ii) the laser irradiation duration controls the number of instructions to be replaced; (iii) the laser irradiation timing controls which instructions are replaced. Section 4 discusses experiments conducted to confirm these points for $N = 1$ case.

4. Demonstration on an ARM Secure Processor

4.1 Experimental Environment

Here, we describe the laser setup and device under test (DUT) used in our instruction replacement experiments. The laser setup was optimized for backside laser irradiation. Note that devices manufactured using recent technology have multiple metal layers that function as shields against laser beams. To avoid the effects of these metal layers, the laser in the experimental setup was fired from the backside of the DUT, i.e., from the silicon substrate side. In this experiment, a smartcard-type ARM SC100 device is used as the DUT because it is easy to expose the silicon substrate of smartcard; thus, such devices are suitable for backside laser irradiation.

4.1.1 Laser Setup

The setup shown in Fig. 2 was assembled for backside laser irradiation. An infrared diode laser (Alphanov PDM+; wavelength 1064 nm) was selected and used to penetrate the silicon substrates of the DUT. The laser was installed in a simple optical system, and fired through a 20× objective lens (SIGMA KOKI PAL-20-NIR-LC00). The laser spot size through the objective lens is approximately 30–40 µm, which is large compared to the diffraction limit (approximately 1 µm), and thus easily achievable with a low quality laser and low quality optical system. The optical system included a specially made infrared ring light for backside imaging, which allowed observation of the circuit layout from the device's silicon substrate side. This enabled easier laser aiming.

With this laser setup, we could control the irradiated point, power, duration and timing laser parameters. The DUT was mounted on an XYZ stage, which allowed us to move the laser irradiation target manually. A stabilized power supply (Matsusada P4K18-2) was employed to control laser power. In addition, pulse signals were generated by a trigger generator to control laser duration and irradiation timing. Note that pulse width and pulse generation timing correspond to laser duration and irradiation timing, which were controlled by Windows OS computer. The trigger generator has a pattern-matching instrument (Riscure icWaves) that generated pulses when the power consumption pattern of the DUT matched a given reference pattern. Such pattern-matching instrument is considered more practical in for a real-world attacker's situation because real-world devices often cannot issue trigger signal for laser irradiation.

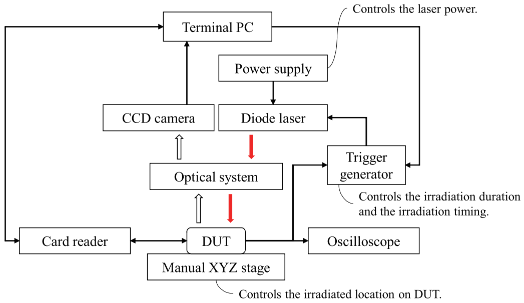

Terminal PC
Controls the laser power.
Power supply
CCD camera
Diode laser
Trigger generator
Optical system
Controls the irradiation duration and the irradiation timing.
Card reader
DUT
Oscilloscope
Manual XYZ stage
Controls the irradiated location on DUT.

Fig. 2 Laser setup assembled for the backside laser irradiation. This setup controls four laser parameters; irradiated location on DUT, irradiation power, irradiation duration and irradiation timing.

Laser fault injection is occasionally considered an impractical threat due to its costs. Commercially available laser attack platforms often cost more than 100k EUR (approximately 150k EURO for the laser platform [8]). To reduce the costs of the experimental setup, we selected and assembled components ourselves. As a result, the total cost of our experimental setup was approximately 30k EUR excluding the oscilloscope and icWaves, which are not included in standard commercial laser platforms.

IcWaves is an optional equipment in the standard commercial laser platforms because icWaves is not necessary for simple fault injection. However, for internal clocking devices such as the DUT in this paper, icWaves is useful to generate the laser shot timing synchronized with the target device. We purchased icWaves at approximately 40k EURO and use it in this paper. The cost comparison including icWaves is the following. Our platform: 30k + 40k = 70k EURO. Commercial platform 100k + 40k EURO. The difference remains about twice including icWaves.

4.1.2 Device Under Test (DUT)

A smartcard-type device is used as the DUT in the experiments conducted to evaluate instruction replacement via laser irradiation of flash memory. It is easy to expose the silicon substrate of such smartcard-type devices by cutting off the IC contact with a knife.

The DUT implements an ARM SC100 core, which is a secure core of the ARM7 family (ARMv4T architecture) and is fabricated according to 90–130 nm process rule.

ARM7 family is a relatively old; however, it has been implemented on a number of devices, such as cellular phones and SIM cards. The ARM7 family applies the von Neumann architecture and a three-stage pipeline, i.e., fetch, decode, and execution. On the DUT, data and programs are stored in the same memory space, and memory instructions and branch instructions take a few cycles, whereas ALU instructions complete within a single cycle. Note that no memory protection or cache mechanisms were employed in these experiments. Here, the fetch stage was targeted for laser irradiation to replace the instructions to be executed. The SC100 core is a secure core; thus, it is conjectured that some anti-tamper mechanism is implemented in this device. However, the flash memory is considered a peripheral unit; therefore, we predict that the proposed instruction replacement method is feasible on the DUT.

The DUT has 64 KB ROM and 4 KB RAM for user space, in which the experimental test programs were stored. An internal clock operates the DUT, and its frequency was estimated as approximately 9 MHz by oscilloscope measurement. Relative to other security mechanisms, the DUT employs data storage scrambling, abnormal voltage detectors (VD) and abnormal clock frequency detectors (FD). The experiments discussed in the following section were all conducted with the VD and FD enabled.

4.2 Experiments

In this section, we demonstrate controllable instruction replacement through laser irradiation of flash memory on the DUT. Here, three laser parameters, irradiated point on the DUT, duration, and timing, were examined to confirm how they affect the instruction replacement.

4.2.1 Effects of Irradiated Points

Figure 3 shows the circuit layout around the flash memory of the DUT. We could easily locate the flash memory because flash memories have noticeable structures and occupy a large part of the entire circuit. The flash memory of the DUT comprises two memory planes (there is a similar plane next to the plane in Fig. 3 (bottom)). Each plane has 32 memory columns. The ARM instruction has a 32-bit instruction length, and it is assumed that specific bit of the instructions is stored in a specific column. The sense amplifiers near the column decoders were irradiated with lasers and examined to determine which irradiated point affects which bit of the instructions.

Program 1 was saved in the DUT's flash memory. This program stores immediate values 0x00 to registers R5–R10 with MOV instructions. Note that registers R5–R10 all have values 0x00 when no errors occur. Here, assume that faulty values 0x01, 0x01, 0x01, 0x00, 0x00, and 0x00 are stored in registers R5–R10 respectively as a result of laser irradiation. In this case, we infer that laser irradiation replaced the first bits of the first three MOV instructions (#3 to #5 in Program 1) with the value "1," because the least significant byte of the MOV instruction indicates an immediate value to be saved in a register (according to the ARM architecture reference manual [9]).

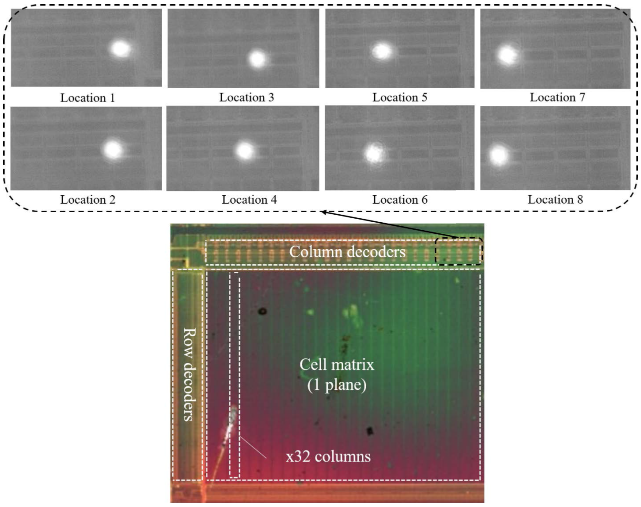


Fig. 3 Circuit layout around the flash memory of the DUT and eight irradiated points (locations) on column decoders.

Program 1: Test program for investigating the effects of laser-irradiated location. Values 0x00 are stored into register R5 to R10.

```
 1: Initialize registers R5 to R10 with 0x00.
 2: NOP x20
 3: MOV R5, 0x00     ;0xe3a05000
 4: MOV R6, 0x00     ;0xe3a06000
 5: MOV R7, 0x00     ;0xe3a07000
 6: MOV R8, 0x00     ;0xe3a08000
 7: MOV R9, 0x00     ;0xe3a09000
 8: MOV R10, 0x00    ;0xe3a0a000
 9: NOP x20
10: Output R5 to R10
```

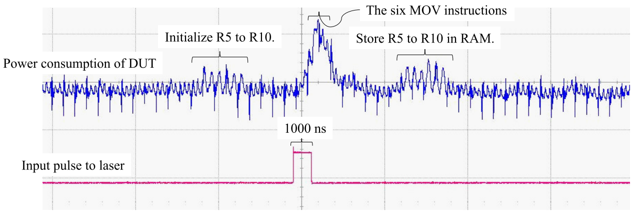


Fig. 4 The DUT's power consumption under laser irradiation.

As shown in Fig. 3, eight locations on the column decoder were irradiated with a laser. Figure 4 shows the changes of the DUT's power consumption caused by the laser irradiation. The laser was driven at 600 mV, and the irradiating duration was 1000 ns, which corresponds to approximately nine instruction cycles. The irradiation timing began before MOV instruction #3 and finished after MOV instruction #7 in Program 1. As shown in Fig. 4, the DUT's power consumption rose at the time of laser irradiation, and

Table 3 Faulty values by laser irradiation (dependent on the eight irradiated locations).

Location	R5	R6	R7	R8	R9	R10
1	0x10	0x00	0x10	0x00	0x10	0x00
2	0x20	0x00	0x20	0x00	0x20	0x00
3	0x80	0x00	0x80	0x00	0x80	0x00
4	0x40	0x00	0x40	0x00	0x40	0x00
5	0x01	0x00	0x01	0x00	0x01	0x00
6	0x02	0x00	0x02	0x00	0x02	0x00
7	0x08	0x00	0x08	0x00	0x08	0x00
8	0x04	0x00	0x04	0x00	0x04	0x00

the laser irradiation affected the timing of executing MOV instructions. Table 3 shows the execution results of Program 1 for each irradiated location. Compared to the normal execution result of 0x00 0x00 0x00 0x00 0x00 0x00, the values in registers R5, R7, and R9 showed a difference of one bit from the original immediate value 0x00. These results indicate that laser irradiation replaced one bit of the immediate value field of the MOV instructions with value “1”; thus, faulty MOV instructions were executed. The positions of the faulty bits of the machine code are dependent on the irradiated location, e.g., irradiated location 1 makes the fifth bit of the 32-bit machine code faulty. While faults were injected in R5, R7 and R9, no faults occurred in R6, R8 or R10. Continuing the experiment, it was found that the eight irradiated locations only affected instructions stored in even memory addresses. By further experiment, we identified the irradiated locations that only affect instructions stored in odd memory addresses.

Next, the same experiment was conducted with a different program, in which the immediate value 0x00 of Program 1 was rewritten as 0xFF. The results demonstrate that no faults occurred on irradiated locations 1–8, which indicates that the replacements by laser irradiation at these locations were stuck-at-1 faults rather than bit-flip faults. Since the targeted immediate values were 0xFF (all bits 1), stuck-at-1 faults on the immediate field of MOV instruction had no effect on the instructions. On the other hand, stuck-at-0 faults were observed at the upper area of each location 1–8. By performing the same experiments on other irradiated locations, we could specify which irradiated locations cause stuck-at-0/1 faults into each bit of 32-bit machine code. Moreover by performing the same experiments to several instructions (ALU, memory operation, and branch instructions), it was observed that the instruction replacements occur except MOV instruction, too. Figure 5 summarizes the results, where irradiated points are represented as a set of coordinates (X, Y). Laser irradiation of the points on the right 16 column decoders of the 32 memory columns made the instructions stored in even memory addresses faulty. Laser irradiation of the locations on the left 16 column decoders of the 32 memory columns made the instruction stored in odd memory addresses faulty. This is conjectured due to the bus-scramble mechanism implemented on the DUT or the two-plane memory structure. The left and right 16 columns of the 32 columns both have 64 points that are sensitive to laser (stuck-at-0 points of each 32 bit and stuck-at-1 points of each 32 bit). The following describes the irradiated points in the right 16 columns because the irradiated points in both right/left 16 columns have the same pattern.

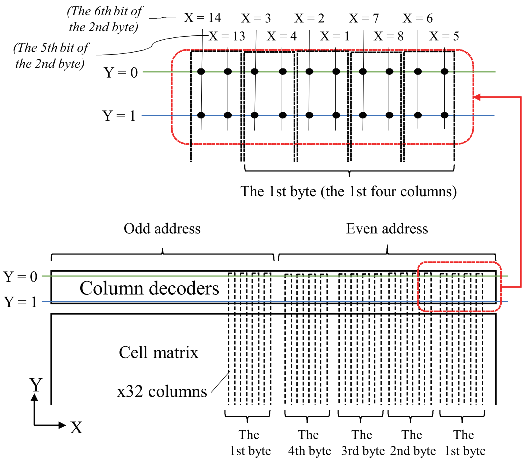


Fig. 5 Relationships among laser-irradiated points and faulty bit positions of the 32-bit instruction. The laser irradiation of coordinates (X, Y) injects stuck-at-Y fault to the Xth bit of the 32-bit instruction. The order of the fault-injected bit for the irradiated points shows the same pattern every 1 byte. This figure shows irradiated points affecting the least significant byte.

For the irradiated points represented as the set of XY coordinates, the Y coordinate of the irradiated points determines that the injected faults are stuck-at-0 or stuck-at-1 faults. There were two Y coordinates sensitive to laser irradiation, i.e., Y = 0 and Y = 1. Laser irradiation of points at Y = 0 injects stuck-at-0 faults to a bit of the 32-bit machine codes. In contrast, laser irradiation of points at Y = 1 injects stuck-at-1 faults to a bit of the 32-bit machine code. The X coordinate of the irradiated points determines the fault-injected bit position of the 32-bit machine code. The 64 irradiated points in the 16 columns demonstrate a pattern, i.e., 16 points are found in each of the four columns, as shown in Fig. 5. Figure 5 (top) shows the 16 points in the first four columns. Laser irradiation of these 16 points caused a stuck-at-0/1 fault in the least significant byte of the 32-bit machine code. Note that laser irradiation of coordinate X = *N* corresponds to the *N*th bit fault of the 32-bit machine code. The same irradiated point pattern can be observed in other four columns. The 16 irradiated points in the second four columns replaced each bit of the second byte of the 32-bit machine code with “0/1,” which was also observed for the third and fourth bytes. In this mean, we could specify the 64 irradiated points that injected stuck-at-0/1 faults to each bit of the 32-bit machine code stored in even memory addresses (as well as the 64 irradiated points affecting the 32-bit machine code stored in odd memory addresses).

Relative to the memory address and bit order, the experimental results differ slightly from the expected results explained in Sect. 3. We consider that this was due to the influence of bus scramble security mechanism implemented

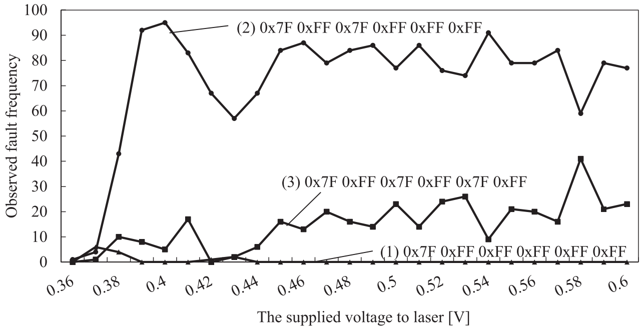


Fig. 6 Relationships between laser power and fault values.

on the DUT and the two-plane memory structure. Even if the relationships among irradiated points and the faulty bit position are complicated, profiling them experimentally enables attackers to inject faults into the target bits.

4.2.2 Effects of Laser Power

To investigate the effects of laser power, an alternate version of Program 1 was saved in the DUT's flash memory, in which values 0xFF were stored in the six registers instead of values 0x00. Coordinate (8, 0) in Fig. 5 was irradiated with a laser a hundred times at a setting of supplied voltage. The supplied voltage was increased from 360 mV to 600 mV in 10 mV increments. Irradiation duration was set to 500 ns at each voltage setting, and the irradiation timing started immediately before the execution of the six MOV instructions.

Figure 6 shows the output values of R5 to R10 after the laser irradiation and their occurrence frequency within 100 irradiation at each voltage setting. Values 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF are outputs of fault-free execution. As shown in the figure, three faulty outputs, (1) 0x7F 0xFF 0xFF 0xFF 0xFF 0xFF, (2) 0x7F 0xFF 0x7F, 0xFF, 0xFF, 0xFF and (3) 0x7F 0xFF 0x7F, 0xFF, 0x7F, 0xFF, were observed during the experiments. According to the earlier experiments in Sect. 4.2.1, laser irradiation of coordinate (8, 0) replaces the 8th bit of machine code stored in even memory addresses with value '0'. The replacements of the 8th bit of machine code correspond to the replacements of the 8th bits of immediate values; the instructions such as MOV regX, 0xFF are replaced with MOV regX, 0x7F, so the faulty values 0x7F are reasonable. For all three kinds of faulty outputs, the faulty value 0x7F appears once per two instructions; this is because the irradiated location (8, 0) affects only the instructions stored in even memory addresses. Whereas the frequency of faulty output (3) increases as the supplied voltage goes up, the frequency of (1) and (2) tend to decrease. These results show that the stronger the supplied power, the longer the effects of laser irradiation; the stronger laser affects more instructions. Since the irradiating duration set to 500 ns corresponds to about 4.5 instruction cycles, it was predicted that the effect of the irradiation that began just before the first MOV instruction did not last until the fifth MOV instruction, when the supplied voltage was low. Also, the frequency of faulty output (2) suddenly went up at approximately 370 mV; afterwards it stayed almost unchanged. This indicates that instruction replacements by laser irradiation occur stably if the supplied voltage is over a certain threshold.

Program 2: Test program for investigating the effects of laser duration. Values 0x01 are added to register R2 255 times.

```
MOV R2, 0x00            ;0xe3a02000
NOP x20
ADD R2, R2, 0x01        ;0xe2822001
| x256 times
ADD R2, R2, 0x01        ;0xe2822001
NOP x20
Output R2
```

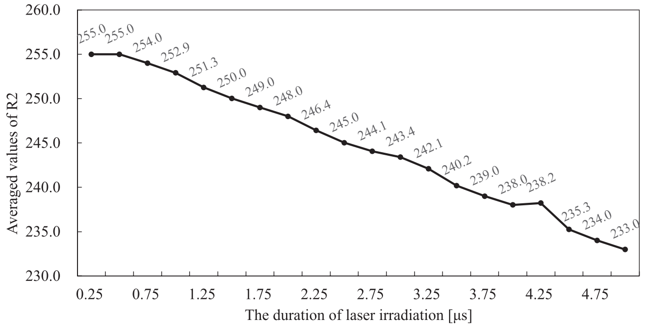


Fig. 7 Relationships between duration of laser irradiation and fault values.

4.2.3 Effects of Laser Duration

To investigate the effects of laser duration, Program 2 was stored on the DUT. This program adds the value 0x01 to register R2 255 times and outputs 0xFF as the value of register R2 when no faults are injected. Here, the machine code of the ADD instruction has an 8-bit immediate value field at the least significant byte. According to the aforementioned result, laser irradiation of the coordinate (1, 0) on the flash memory should replace ADD R2, R2, 0x01 instructions with ADD R2, R2, 0x00 instructions. Note that a longer laser duration is expected to replace more instructions (the longer the irradiation time, the lower the value of register R2). The laser was supplied with 600 mV, and the duration was increased from 250–5000 ns in 250 ns increments, which corresponds to approximately two instruction cycles. For each laser duration, the flash memory was irradiated 100 times, and the output of Program 2 was recorded each time. The obtained outputs differ from each other because the irradiation timing fluctuated somewhat due to internal clock jitters. As a representative value, the average values of each 100 executions were plotted.

The experiment results are shown in Fig. 7. Value 255 was observed at 250 ns and 500 ns; thus, no faults were injected because the laser irradiation began with the NOP instructions (#2 in Program 2) and finished before the ADD instructions. The least significant bit of the NOP instructions is 0; thus, laser irradiation of coordinate (1, 0) did not affect the NOP instructions. For settings above 750 ns, the output values of Program 2 decreased linearly by approximately 1 every 250 ns. Note that 250-ns duration corresponds to two instruction cycles. By the increasing laser duration by 250 ns, two more ADD instructions were fetched during irradiation. One of the two additional ADD instructions was stored in an odd memory address; thus, it was unaffected by laser irradiation of coordinate (1, 0). As a result, it was observed that increasing laser duration by 250 ns reduced the output of Program 2 by approximately 1.

We have confirmed that laser duration can control the number of instructions to be replaced. The ease by which faults can be injected in multiple instructions by increasing laser duration is an advantage of the proposed method. This feature might compromise the security of existing anti-fault countermeasures that assume attackers will not inject multiple faults.

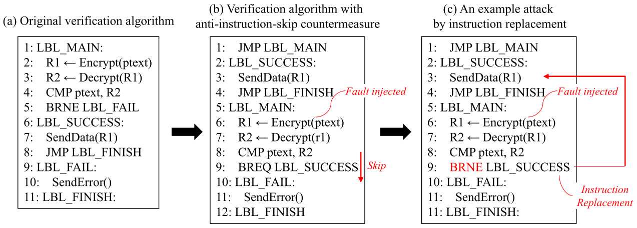
4.2.4 Effects of Irradiation Timing

To investigate the effects of irradiation timing, Program 1 was stored on the DUT, similar to the experiment discussed in Sect. 4.2.1. Under fault-free execution, this program outputs the values 0x00 0x00 0x00 0x00 0x00 0x00. Here, the laser duration was 50 ns, which corresponds to less than one clock cycle. Therefore, this laser irradiation affects only one instruction per shot. The irradiated point and laser power were (1, 1) and 600 mV, respectively;. It was expected that Program 1 would output 0x01 (as a fault value) rather than 0x00.

At irradiation timing t_0, faulty outputs 0x01 0x00 0x00 0x00 0x00 0x00 were observed. The irradiation timing was increased from t_0 in 10-ns increments. As a result, three faulty outputs 0x01 0x00 0x00 0x00 0x00 0x00; 0x00 0x00 0x01 0x00 0x00 0x00; and 0x00 0x00 0x00 0x00 0x01 0x00 were obtained at the irradiation timing of t_0 to t_0 + 50 ns, t_0 + 200 ns to t_0 + 250 ns, and t_0 + 400 ns to t_0 + 450 ns. These results demonstrate that we can control desired timing of instruction replacement by irradiation timing.

5. Instruction Replacement Attack on Existing Countermeasures against Instruction Skip

Many existing countermeasures against instruction skips are based on some sort of redundancy. Figure 8 a countermeasure based on algorithm-level redundancy. This countermeasure encrypts a plaintext and decrypts the output for verification. The decrypted result is compared to the plaintext, and the countermeasure calls the output function only if no fault is detected. Simple implementations of such verification countermeasures are known to be vulnerable to instruction skip attacks. Figure 8(a) shows an example of a simple implementation where skipping the branch instruction (line 5) leads to faulty ciphertext output. Endo et al. proposed a modified version of this verification algorithm (Fig. 8(b)), called "defaultfail", to prevent instruction skip from unexpectedly outputting the ciphertext [10]. Note that defaultfail countermeasure locates the output function at an upper address than the branch instruction, where no data are output if the branch instruction is skipped.



(a) Original verification algorithm

```
1: LBL_MAIN:
2:    R1 ← Encrypt(ptext)
3:    R2 ← Decrypt(R1)
4:    CMP ptext, R2
5:    BRNE LBL_FAIL
6: LBL_SUCCESS:
7:    SendData(R1)
8:    JMP LBL_FINISH
9: LBL_FAIL:
10:    SendError()
11: LBL_FINISH:
```

(b) Verification algorithm with anti-instruction-skip countermeasure

```
1:    JMP LBL_MAIN
2: LBL_SUCCESS:
3:    SendData(R1)
4:    JMP LBL_FINISH
5: LBL_MAIN:
6:    R1 ← Encrypt(ptext)
7:    R2 ← Decrypt(r1)
8:    CMP ptext, R2
9:    BREQ LBL_SUCCESS
10: LBL_FAIL:
11:    SendError()
12: LBL_FINISH
```

(c) An example attack by instruction replacement

```
1:    JMP LBL_MAIN
2: LBL_SUCCESS:
3:    SendData(R1)
4:    JMP LBL_FINISH
5: LBL_MAIN:
6:    R1 ← Encrypt(ptext)
7:    R2 ← Decrypt(R1)
8:    CMP ptext, R2
9:    BRNE LBL_SUCCESS
10: LBL_FAIL:
11:    SendError()
11: LBL_FINISH:
```

Fig. 8 Example instruction replacement attack to the algorithm-level countermeasure against instruction skip [10].

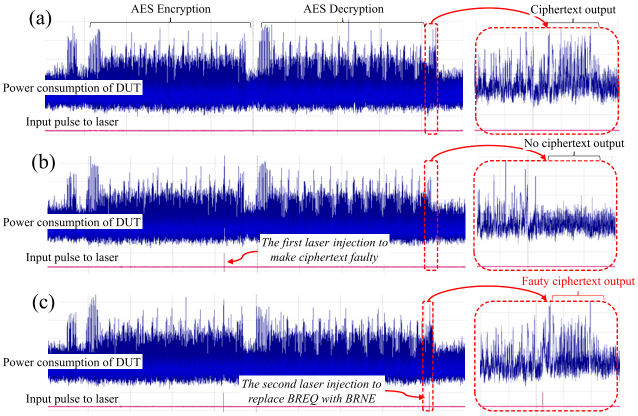


Fig. 9 Results of demonstration attack to defaultfail countermeasure. (a) When no faults occur; correct ciphertext is output after AES encryption and AES decryption. (b) When the first fault is injected during AES encryption;nNo ciphertext is output because the ciphertext becomes faulty and verification fails. (c) When the first fault is injected followed by the second fault injected to BREQ instruction; verification fails, but the faulty ciphertext is output because BREQ is replaced with the BRNE instruction.

Instruction replacement can attack such defaultfail countermeasure by replacing BREQ instructions (an instruction that branches when values are equal) with BRNE instructions (an instruction that branches when values are not equal). To demonstrate this attack, we implemented the defaultfail algorithm on the DUT, where AES was used for both encryption and decryption. Note that the Hamming distance between BREQ LBL_SUCCESS and BRNE LBL_SUCCESS is 1 in the ARM instruction set; thus, 1-bit instruction replacement with our laser setup could inject such replacements of branch instructions. The experimental results are shown in Fig. 9. Figure 9(a) shows the DUT's power consumption when no fault was injected. The correct ciphertext was output after verification. Figure 9(b) the shows DUT's power consumption when the first fault (to make ciphertext faulty) was injected. Here, no ciphertext was output from the DUT. This was also observed from

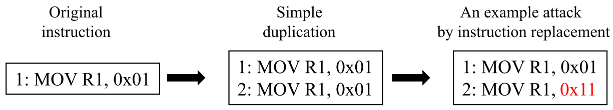


Fig. 10 Example instruction replacement attack to the instruction-level countermeasure against instruction skip.

the DUT's power consumption, i.e., the power consumption after AES decryption differs from that shown in Fig. 9(a). Next, the second fault (i.e., replace BREQ with the BRNE instruction) was injected subsequent to the first fault. The result is shown in Fig. 9(c). It was found that the ciphertext was output even though the first fault was injected. Since BREQ was replaced with BRNE, the ciphertext was output if and only if the ciphertext included faults. The result shown in Fig. 9(c) demonstrates that we were able to obtain faulty ciphertexts successfully.

Figure 10 shows examples of a countermeasure based on instruction-level redundancy. Moro et al. proposed a countermeasure against instruction skip by duplicating instructions [6], and they applied this countermeasure to Thumb-2 instruction set. The proved formally that the countermeasure is secure against an attacker that can inject only a single instruction skip. As shown in Fig. 10, instruction replacement can easily attack this countermeasure. Moreover, a more sophisticated countermeasure, i.e., a combination of instruction duplication and verify computation of duplicated instructions, has been proposed by Barenghi et al. [11]. Note that instruction replacement can also attack this countermeasure through a combination of the aforementioned replacement techniques.

6. Discussion: Controllability and Limitation of the Proposed Attack

6.1 Controllability for the Replaced Bit Position

The experiments in Sect. 4.2.1 revealed where the location of irradiated locations control which bit of instruction is replaced, over each bit of entire 32-bit word. Assume that attackers can conduct such exploration experiments in advance. Controlling the laser irradiated location, the attackers can change one bit of instructions running on the attack target. The proposed method is an attack that replaces data to be reading from flash ROM rather than instructions (Many embedded systems use flash ROMs for instruction storage). Therefore, we suppose that the effects of the laser irradiation are not dependent on instructions before replacements and that the proposed method can replace every instruction to be fetched from flash ROM at the same manner. This is found out from the results of the experiments in Sect. 4.2.1, Sect. 4.2.3., and Sect. 5. Three different instructions MOV, ADD, and BREQ were attacked in the respective experiments, and expected faults were injected to three of them).

6.2 Controllability for the Number of Instruction Replacements

From the results in Sect. 4.2.3, we found that the longer laser duration causes the instruction replacement over multiple clock cycles. Therefore, the higher clock frequency, the more difficult to inject an instruction replacement fault to one desired instruction. Although the target device runs with the low frequency of 9 MHz as you pointed out, we suppose that the proposed method is applicable to the devices running with high frequency. Let T_{clk} and $T_{duration}$ be the clock period of target device and be the laser duration. In order not to cause more than one instruction replacement, to satisfy the next expression is required.

$$T_{duration} < 2T_{clk}. \quad (1)$$

Our laser platform can generate 2 ns laser pulse at minimum; thus, we suppose that our laser platform can attack the devices running with up to 1GHz.

6.3 Limitation of the Number of Bits Possible to Be Replaced

The experiments conducted in this paper use mono-spot laser station. As mentioned in Sect. 3, this condition has a limitation that the replacement of Hamming distance more than one, between instruction before/after replacement, is impossible. Moreover, another bit replacement after a bit replacement is supposed also difficult since moving laser during the target running to another irradiated point is referred as a hard task. However, even the replacement of Hamming distance one leads to a big threat such as the branch instruction replacement in Sect. 5. Multi-spot laser can irradiate flash ROM with multiple points, thus, it can replace multiple bit of an instruction and different bits of an instruction and another instruction. However, multi-spot laser platform is highly expensive, for example, commercial double-spot laser at least exceeds 200k EURO. Multi-bit replacement is not impossible but considered not practical.

6.4 Attack on Other Devices

We consider that the proposed method is applicable to other processor architectures because the proposed method replaces instructions by perturbing the operation of the sense amplifiers in flash ROM. Furthermore, while process shrink is referred as making laser fault injection difficult, NOR-type flash ROM, which is used for program storage in many embedded systems, reaches limitation of the shrink (See cypress 2019 flash ROM road map page 4 [12]. Small process less than 45 nm is not presented). Therefore, the proposed method is supposed applicable to the cutting-edge devices.

7. Conclusion

This paper has shown the following three contributions:

(I) Concept of controllable instruction replacement, (II) Demonstration of the instruction replacement concept on an ARM secure processor, and (III) Instruction replacement attacks that circumvent an existing countermeasure against instruction skip.

Through a series of experiments, we have demonstrated that instruction replacement is feasible on an actual ARM device. Many software-based countermeasures against fault attacks attempt to prevent instruction skip faults: however, instruction replacement faults could compromise the security of such countermeasures. In fact, other than this paper, (uncontrollable) instruction replacement faults have been observed on some devices through various fault injection means, these observations can possibly compromise the existing countermeasures. Note that the instruction replacement fault is a common and general threat to embedded devices; thus, the countermeasures against instruction replacement attacks are required in future.

Acknowledgments

The target device for the laser experiments in this paper was provided by the Information-technology Promotion Agency, Japan (IPA). We thank the staff of the IPA for their support

A part of this research was supported by SECOM Science and Technology Foundation. We thank them for their support.

This work was supported by the Cabinet Office (CAO), Cross-ministerial Strategic Innovation Promotion Program (SIP), “Cyber Physical Security for IoT Society” (funding agency: NEDO).

References

[1] J. Balasch, B. Gierlichs, and I. Verbauwhede, “An in-depth and black-box characterization of the effects of clock glitches on 8-bit MCUs,” 2011 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2011, pp.105–114, Tokyo, Japan, Sept. 2011.

[2] B. Yuce, N.F. Ghalaty, H. Santapuri, C. Deshpande, C. Patrick, and P. Schaumont, “Software fault resistance is futile: Effective single-glitch attacks,” 2016 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2016, pp.47–58, Santa Barbara, CA, USA, Aug. 2016.

[3] N. Moro, A. Dehbaoui, K. Heydemann, B. Robisson, and E. Encrenaz, “Electromagnetic fault injection: Towards a fault model on a 32-bit microcontroller,” CoRR, vol.abs/1402.6421, 2014.

[4] L. Riviére, Z. Najm, P. Rauzy, J. Danger, J. Bringer, and L. Sauvage, “High precision fault injections on the instruction cache of ARMv7-M architectures,” CoRR, vol.abs/1510.01537, 2015.

[5] E. Trichina and R. Korkikyan, “Multi fault laser attacks on protected CRT-RSA,” 2010 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2010, pp.75–86, Santa Barbara, California, USA, Aug. 2010.

[6] N. Moro, K. Heydemann, E. Encrenaz, and B. Robisson, “Formal verification of a software countermeasure against instruction skip attacks,” J. Cryptogr. Eng., vol.4, no.3, pp.145–156, 2014.

[7] A. Barenghi, G. Bertoni, L. Breveglieri, M. Pellicioli, and G. Pelosi, “Low voltage fault attacks to AES,” HOST 2010, Proc. 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp.7–12, Anaheim Convention Center, California, USA, June 2010.

[8] O.M. Guillen, M. Gruber, and F. De Santis, “Low-cost setup for localized semi-invasive optical fault injection attacks,” Constructive Side-Channel Analysis and Secure Design, S. Guilley, ed., pp.207–222, Springer International Publishing, Cham, 2017.

[9] ARM, ARM Architecture Reference Manual, https://cs.nyu.edu/courses/spring18/CSCI-GA.2130-001/ARM/arm_arm.pdf

[10] S. Endo, N. Homma, Y. Hayashi, J. Takahashi, H. Fuji, and T. Aoki, “A multiple-fault injection attack by adaptive timing control under black-box conditions and a countermeasure,” COSADE, Lecture Notes in Computer Science, vol.8622, pp.214–228, Springer, 2014.

[11] A. Barenghi, L. Breveglieri, I. Koren, G. Pelosi, and F. Regazzoni, “Countermeasures against fault attacks on software implemented AES: Effectiveness and cost,” WESS, p.7, ACM, 2010.

[12] Cypress, Cypress Roadmap: Flash Memory Q2 2019, https://www.cypress.com/file/206951/download

Junihci Sakamoto received his B.E. and M.I.S degrees from Yokohama National University, Japan, in 2015, and 2017, respectively. He is currently pursuing his Doctorate degree. He has engaged in various researches regarding information security, including the methodology of secure implementation of the cryptographic algorithms side-channel attacks to pairing computation, and laser-based fault attack.

Daisuke Fujimoto received B.E., M.E., and Ph.D. degrees from Kobe University, Japan, in 2009, 2011 and 2014, respectively. He is currently an assistant professor at the Graduate School of Information Science, Nara Institute of Science and Technology, Nara, Japan. He is also a visiting assistant professor in the Institute of Advanced Sciences, Yokohama National University. His research interests include hardware security and the implementation of security cores. He is a member of IEEE and IEICE.

Tsutomu Matsumoto is a professor at the Faculty of Environment and Information Sciences, Yokohama National University and directing the Research Unit for Information and Physical Security at the Institute of Advanced Sciences. He received Doctor of Engineering from the University of Tokyo in 1986. Starting from Cryptography in the early eighties, he has opened up the field of security measuring for logical and physical security mechanisms. Currently he is interested in research and education of Embedded Security Systems such as IoT Devices, Network Appliances, Mobile Terminals, In-vehicle Networks, Biometrics, Artifact-metrics, and Instrumentation Security. He is serving as the chair of the IEICE Technical Committee on Hardware Security, the Japanese National Body for ISO/TC68 (Financial Services), and the Cryptography Research and Evaluation Committees (CRYPTREC) and as an associate member of the Science Council of Japan (SCJ). He was a director of the International Association for Cryptologic Research (IACR) and the chair of the IEICE Technical Committee on Information Security. He received the IEICE Achievement Award, the DoCoMo Mobile Science Award, the Culture of Information Security Award, the MEXT Prize for Science and Technology, and the Fuji Sankei Business Eye Award.