Characterization of EM faults on ATmega328p

Arthur Beckers*, Josep Balasch*, Benedikt Gierlichs*, Ingrid Verbauwhede*,
Saki Osuka†, Masahiro Kinugawa†, Daisuke Fujimoto†, Yuichi Hayashi†
†Nara Institute of Science and Technology, 8916-5 Takayama-cho, Ikoma, Nara 630-0192, Japan
*imec-COSIC, KU Leuven Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
*firstname.lastname@esat.kuleuven.be

***Abstract*—We investigate the effects of EM pulses on an ATmega328p 8-bit microcontroller. We establish which areas of the chip are sensitive to EM pulse injection and describe the fault model for these sensitive areas. Furthermore, we compare our results to those of a previous study, which examined the effects of laser fault injection on the same device.**

I. Introduction

Fault attacks are well-known techniques that target security functionalities implemented in electronic devices [1]. They involve an adversary capable of causing (typically transient) circuit-level errors by actively tampering with the device, or with its close environment. Inducing a fault at a critical stage of a security-sensitive computation can lead to exploitable errors at implementation-level. Particularly for micro-controllers, errors can affect both the program flow (e.g. skipping or modifying instructions) and the data flow (e.g. flipping, setting or resetting one or more bits in intermediate variables). Depending on the actual *fault model*, fault attacks can be used to bypass security features (e.g. password checks) or to infer sensitive information (e.g. secret keys from cryptographic implementations [2], [3]).

Fault injection mechanisms are typically categorized in two groups: *non-invasive* methods can be used without package modifications, while *semi-invasive* methods require line of sight to the circuit die. Classical fault injection mechanisms are non-invasive. They often perform a short-time manipulation of external signals, for instance clock glitches [4] or voltage spikes [5]. Consequently, they have a global effect on the circuit. In contrast, semi-invasive attacks enable fine-grained spatial resolution and can target isolated components of a circuit (e.g. CPUs, memories, or arithmetic co-processors). The most popular semi-invasive fault attack is an optical attack [6] that uses coherent light produced by a laser source to induce a faulty behavior to the transistors. A popular type of non-invasive attack are electromagnetic (EM) attacks [7] that direct intense and short EM pulses to the circuit in order to cause sudden current flows in its power/ground networks.

In this work we investigate the effects of EM pulses on the flash memory of an ATmega328p 8-bit microcontroller. The goal of our study is to obtain a fault model that captures the characteristics and effects of the injected errors. Deriving accurate fault models is not only necessary to uncover potential attack vectors, but also to develop sound mitigation strategies. Yet the characterization of fault effects is far from trivial. In fact, the fault model is intimately linked to the fault injection mechanism used by the adversary and to the characteristics of the target device. Therefore, fault models are often studied in a case-by-case basis.

Related Work. The susceptibility of embedded micro-controllers and FPGAs to EM pulse fault injection has been investigated in [8]. The main conclusions of this work are: first, that the induced faults are local. It is possible to affect only a relatively small area of the device under test (DUT); second, that the induced faults are not frequency dependent. The authors were only able to inject faults into the device when the pulse was generated close to the clock edge, which points at a violation of the setup and hold times. A similar study has been presented in [9], which performed a more in depth study of the type of faults induced into a 32-bit ARM Cortex-M3 micro-controller. The experiments show that only data read from the flash memory of the micro-controller is susceptible to EM pulse injection. The fault model they describe is the "set to 1" fault model. The EM-pulse injection leads to some bits of the read data to be set to 1.

The susceptibility of the ATmega328p to laser fault injection has been recently investigated in [10]. Here the authors showed that the readout of data from the ATmega328P's flash memory has a "set to 0" fault model. The position of the laser relative to the DUT determined which bits of the read out data were set to zero. With a high degree of precision and a 100% repeatability one or more bits of the loaded data could be set to 0. For their experiments the authors used a 1064 nm laser to perform through substrate laser fault injection.

Additionally, the effects of clock glitches on an ATmega163 micro-controller have been investigated in [4].

Our contributions. We investigate the effects of EM pulses on the flash memory of an ATmega328p 8-bit microcontroller. We establish which areas of the chip are sensitive to EM pulse injection and describe the fault model for these sensitive areas.

The main reason for choosing the ATmega328P as our target is a previous [10] study on its susceptibility to laser fault injection. This enables us to perform a comparative study between the two different fault injection methods on the same device.

II. Experimental setup

We describe our fault injection setup as well as the methodology followed to characterize and understand the effects of injecting a fault.

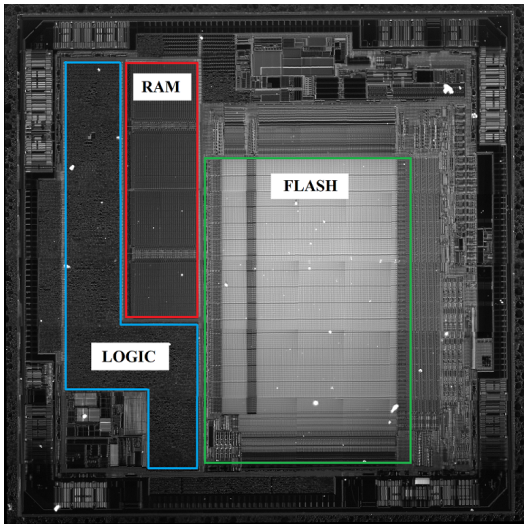


Fig. 1. Trough substrate image ATmega328P.

A. ATmega328p

The target device is the ATmega328p [11]. It is an 8-bit AVR micro-controller with 32kB of flash memory, 1kB of EEPROM and 2kB of RAM. Figure 1 shows a through substrate image of the ATmega328p. The different components that make up the microcontroller can be clearly distinguished. The ATmega328p has a two stage pipeline with fetch and execute stages. For our experiments we set the clock frequency of the DUT to 8MHz. Running the DUT at a lower frequency than the maximal 16MHz allows us to more easily target individual instructions during the characterization.

The ATmega328p is used in the popular Arduino UNO platform, which is widely available and well documented. The ATmega328p is placed in a DIP socket on the board which makes it easy to swap out the DUT should it be damaged by the EM pulse injection.

One of the main advantages of EM pulse injection is that it does not require decapsulation of the target device. The EM-pulse generated by the injection setup can propagate unhindered through the plastic packaging of the device. Decapsulation however does have the advantage that it allows the injection probe to be placed closer to the target. This semi-invasive method increases the resolution of the fault injection setup and reduces the power needed to fault the device. Therefore we opted to depackage the DUT from the backside exposing the die of the DUT. This allows us to place the injection probe as close as 500 μm from the metal layers of the DUT.

B. EM pulse injection setup

EM pulse injection setups generally consist of two main components: the pulse generator and the injection probe. A good pulse generator is capable of generating a high voltage pulse with a short rise time. The shape of the pulse determines the size of the induced current and the time resolution of the EM pulse injection. The total rise time will determine the time resolution of the probe, while the instantaneous rise time will determine the size of the induced current.

For our characterization we make use of the Langer EM fault injection setup [12]. It consists of the BPS 202 burst power generator (BPS) and the ICI HH500-50 magnetic field pulse source. The power generator can supply up to 500V to the magnetic probe. The switching circuitry is situated inside the magnetic field pulse source itself and is capable of delivering pulses with a 2 ns rise time. The short rise time allows us to target a single clock cycle of the ATmega328P. The EM-probe consists of copper wiring wound around a conical ferrite core. The diameter of the probe tip is 500 μm. This is a common probe shape for generating or measuring magnetic fields. The probe is mounted on an XYZ stepper table with a 20 μm resolution in order to accurately position the probe relative to the DUT.

TABLE I
TEST CODE.

Cycle	Instruction
1	sbi 0X0B, 7 // trigger
2	nop
3	nop
4	target
5	nop
6	nop

C. Methodology

For our characterization we follow a similar approach as in [10]. We execute a target assembly instruction surrounded by NOPs on the DUT (see table I). Before executing the target code we set the DUT's working registers to a known state. After the EM pulse injection we read back the content of these registers and determine whether or not a fault was injected.

An overview of the entire EM pulse injection setup can be seen in Figure 2. To start the pulse injection the PC gives a start command to the DUT which runs the target code. Before the target code is executed the DUT sends a trigger signal to the oscilloscope which in turn triggers the BPS. The power, delay and orientation of the injected pulse are programmed into the BPS by the PC. After the pulse injection the DUT sends its register content to the PC. We profile the device by stepping over the entire surface of the chip with 100 um steps. The die of the ATmega328p measures three by three millimeters. At each position we repeat the procedure 100 times. Besides varying the location of our EM-pulse injection we also vary the timing of the pulse injection. At every location we increment the delay of our EM-pulse injection with 10 ns increments, making sure we inject pulses in the cycle right before and after the fetch and execute of our target instruction. By comparing the content of the different working registers with their expected values we can determine the type of fault induced into the DUT by the EM pulse.

III. EXPERIMENTAL RESULTS

For the initial profiling of the ATmega328p we injected EM pulses into the DUT while running assembly code structured

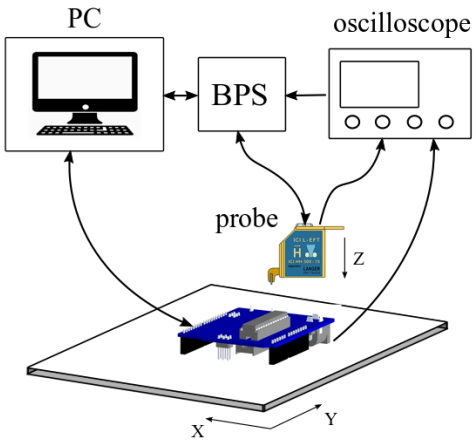


Fig. 2. Experimental setup.

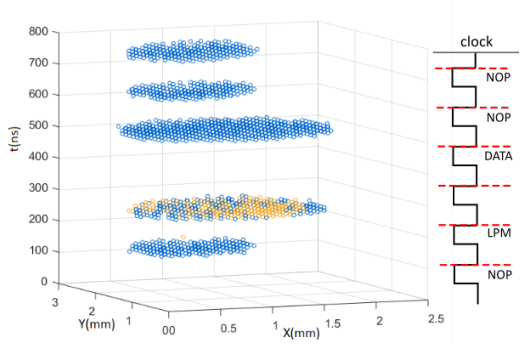
as in Table I using different target instructions. The target instructions performed different arithmetic operations, reading from and writing to memory. The goal of this initial profiling step was to determine the regions of the chip sensitive to EM pulse injection. We found that only the area containing flash memory was sensitive. Therefore we will focus on faulting the flash memory in the remainder of the paper.

Since only the flash memory is sensitive to EM pulse injection we can determine the fault model of the device using the LPM instruction. This three cycle instruction reads data from flash memory. The actual data transfer over the bus happens during the second cycle of its execution. Faulting the read out of data has the advantage that the program flow remains unaltered. Another advantage is that we can set the data stored in the memory to a value of our choosing.

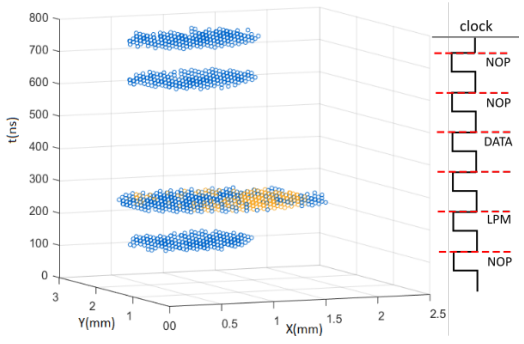
We performed two experiments. In the first one we filled part of the flash memory with all zeros (0x00). As target instruction (Table I) we use the LPM instruction that reads from one of these memory locations and stores the result in a working register. We use the methodology described in II-C to profile the DUT. We thus scan the entire chip with both time and location as a variable. The results of this profiling can be seen in Figure 3a. Each blue dot represents a combination of location and time for which we can induce a fault into the DUT. The orange dots represent crashes of the device: the device enters an unknown state from which it can only recover by applying a hard reset. During each clock cycle either instructions or data is fetched from flash memory. On the righthand side of Figure 3 the information transfered from flash memory during each clock cycle is depicted. We can clearly see that the faults have a tendency to cluster in the time domain. These clusters lie at 125 ns intervals from each other, which corresponds to the length of one clock cycle. This indicates that we can only fault an instruction at a clock edge. This behaviour corresponds to what was previously described in [8].

In Figure 3a the first cluster of faults at 125 ns corresponds to the fetching of the NOP before the fetching of the target instruction. The second cluster at 250 ns corresponds to the fetching of the LPM instruction. After the second cluster there is a clear gap at 375 ns where no faults are induced. This is the first cycle of the execution of the LPM instruction during which no data that impacts the data or program flow is transmitted over the flash memory bus. The third cluster at 500 ns corresponds to the reading of the data from flash. The two remaining clusters again correspond to the fetching of the NOPs that come after the target instruction.

In a second experiment we filled a part of flash memory with all ones (0xff) instead of zeros. We again followed the methodology of section II-C to profile the chip. The result can be seen in Figure 3b. We notice that at 500 ns, the moment at which the data is fetched from memory, no faults are injected when we load 0xff from flash memory. In the previous experiment, loading 0x00 from memory resulted in a large cluster of faults at this point in time. This leads us to conclude that the reading from flash memory of an ATmega328p has a "set at 1" fault model when EM pulses are injected. Bits read from flash memory can be set to one but bits already set at one can not be reset into a zero. A similar fault model was found in [9] when injecting EM faults into the flash memory of an ARM microcontroller.



(a) Loading all zeros



(b) Loading all ones

Fig. 3. Fault distributions in time and space with LPM as target instruction.

Figure 4 shows the sensitive regions of the chip when reading all zeros from flash. A sensitive region is a region where at least one bit of the read data is faulted. The sensitive spots on the DUT are marked in color ranging from blue to red. Blue meaning that at most one of the bits is set to one.

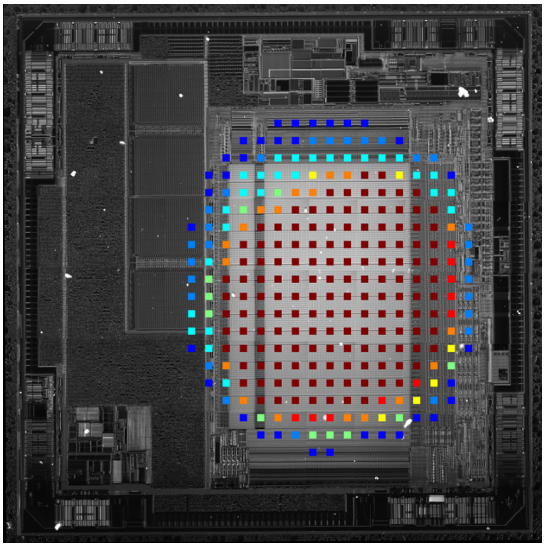


Fig. 4. Sensitive region when loading from an even memory address.

Red on the other hand indicates that a fault can be introduced that puts all bits at one.

EM fault injection is hard to control, at least with our setup. If we repeat an injection keeping all parameters the same, the observable fault may be different. Moreover, changing even only one of the parameters a little bit may lead to a completely different observable fault.

IV. Fault model comparison

In this section we will compare the fault model for EM fault injection to the one for laser fault injection. The authors of [10] showed that laser fault injection is able to induce "set to 0" faults into data read from flash memory. They were able to reset individual or multiple bits with high accuracy and 100% repeatability.

Just as with laser fault injection only the flash memory of the ATmega328p is sensitive to EM pulse injection. While laser fault injection has a "set to 0" fault model, the faults induced by EM pulse injection have a "set to 1" fault model. One of the main advantages of laser fault injection over EM fault injection is its repeatability. With EM fault injection rather than having a 100% repeatability there is on every location only a probability a certain fault is injected. With laser fault injection an attacker is also capable of targeting a single or multiple bits. With EM pulse injection on the other hand the attacker has very little control over which bits are faulted.

With EM pulse injection the data read from the flash memory can only be faulted when the pulse is injected around a clock edge. With laser fault injection one is able to fault the reading of flash data during the entire clock cycle. The ability to inject faults during the entire clock cycle reduces the timing precision needed in the laser setup to mount an attack.

V. Conclusion

Our study confirms the main findings of previous work regarding EM pulse injection on micro-controllers. It appears that only the reading of data from the flash memory can be faulted. Furthermore, it appears that the fault model is "set to 1". The effect is transient, i.e. the actual values in flash memory are not changed. Only the reading is affected.

In comparison to laser fault injection, EM fault injection requires less sophisticated equipment. On the other hand, laser fault injection seems to have clear advantages with respect to accuracy and repeatability. In addition, the fault models differ. Hence there is no "best" method for all situations.

Our study also informs the design of countermeasures. In particular, it seems clear that the different fault models for EM and laser are due to different injection mechanisms, i.e. different parts of the circuit are affected. Hence sensors aiming to detect a laser attack may fail to detect an EM attack and vice versa.

Acknowledgment

This work was supported in part by the Research Council KU Leuven: C16/15/058 and through the Horizon 2020 research and innovation programme under Cathedral ERC Advanced Grant 695305. Additionally this work has been partially supported by FWO project VS06717N in collaboration with JSPS.

References

[1] H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan, "The sorcerer's apprentice guide to fault attacks," *Proceedings of the IEEE*, vol. 94, no. 2, pp. 370–382, Feb 2006.

[2] D. Boneh, R. A. DeMillo, and R. J. Lipton, "On the importance of checking cryptographic protocols for faults (extended abstract)," in *EUROCRYPT '97*, ser. LNCS, W. Fumy, Ed., vol. 1233. Springer, 1997, pp. 37–51.

[3] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Advances in Cryptology — CRYPTO '97*, K. S. Burton, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 513–525.

[4] J. Balasch, B. Gierlichs, and I. Verbauwhede, "An In-depth and Black-box Characterization of the Effects of Clock Glitches on 8-bit MCUs," in *FDTC 2011*, L. Breveglieri, S. Guilley, I. Koren, D. Naccache, and J. Takahashi, Eds. IEEE Computer Society, 2011, pp. 105–114.

[5] J. Schmidt and C. Herbst, "A practical fault attack on square and multiply," in *FDTC 2008*, L. Breveglieri, S. Gueron, I. Koren, D. Naccache, and J. Seifert, Eds. IEEE Computer Society, 2008, pp. 53–58.

[6] S. P. Skorobogatov and R. J. Anderson, "Optical fault induction attacks," in *CHES 2002*, ser. LNCS, B. S. K. Jr., Ç. K. Koç, and C. Paar, Eds., vol. 2523. Springer, 2002, pp. 2–12.

[7] P. Maurine, "Techniques for em fault injection: Equipments and experimental results," in *2012 Workshop on Fault Diagnosis and Tolerance in Cryptography*, Sept 2012, pp. 3–4.

[8] S. Ordas, L. Guillaume-Sage, and P. Maurine, "Em injection: Fault model and locality," in *2015 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, Sept 2015, pp. 3–13.

[9] 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. [Online]. Available: http://arxiv.org/abs/1402.6421

[10] D. S. V. Kumar, A. Beckers, J. Balasch, B. Gierlichs, and I. Verbauwhede, "An in-depth and black-box characterization of the effects of laser pulses on atmega328p," in *CARDIS 2018*, ser. LNCS, B. Bilgin and J. Fischer, Eds. Springer, 2018.

[11] Microchip, "Atmega328p," https://www.microchip.com/wwwproducts/en/ATmega328P, July 2018.

[12] Langer EMV, "ICI 01 L-EFT set," https://www.langer-emv.de/en/product/ic-side-channel-analysis/94/ici-01-1-eft-set-ic-em-pulse-injection-langer-pulse/821, December 2018.