Electro Magnetic Fault Injection

Research Project
Universiteit van Amsterdam
System and Network Engineering (MSc)

Class of 2011-2012

Sebastian Carlier (sebastian.carlier@os3.nl)

January, 2012

Abstract

This paper researches electromagnetic fault injection, as a viable form of attack on smartcards and embedded systems. We show that it is possible to influence the execution of instructions on an ATMega 163+ smartcard and an LPC1769 micro-controller. The paper explains the method followed during testing and focuses on the parameters involved in the process. Mainly the ones that influence the electromagnetic burst emitted from the electromagnetic probe's coil (ie. attacking device): distance to the attacked device, the size of the coil, power supplied to the attacking device and the influence of a ferrite core were tested. Temporal parameters, relative to the execution of instructions on the target, are also described. By comparing the results produced by different settings of each parameter it is shown that minimal distance should be kept, as it is the most influential parameter. A 4mm coil influences both targets the most and a higher amplitude of the voltage spike introduces more glitched outputs. However the results show that it is advisable to use a smaller coil and an even higher amplitude, because a bigger coil resets the target more often, due to its area of influence. 23% of the smartcard's executions and 9% of the embedded system's executions were glitched with optimal settings of the parameters.

Contents

1 Introduction

This paper describes Electro Magnetic Fault Injection: a side channel attack that introduces faults to a system by influencing it with a flux of the magnetic field. Side channel attacks are a type of attack that do not target a cryptographic algorithm directly, but the underlying hardware on which the algorithm is executed. They are used when the algorithm is too strong to attack with more traditional methods like brute forcing[8]. An EMFI attack can introduce a fault in the original computation by disrupting its execution flow. With a certain amount of unique faulty outputs the secret key can be revealed through differential fault analysis[9]. EMFI is similar to optical glitching attacks that target smart cards or embedded systems, but it introduces certain benefits. First of all, the target chip does not need to be decapsulated which is the case with optical attacks. This process damages the hardware. Secondly, some chips have security measures preventing them from being susceptible to optical fault injection attacks which is not the case with EMFI.

1.1 Scope

The main focus of this research is to show that EMFI is feasible. The research focuses on the parameters of the attack that influence the amplitude of voltage perturbation on the target. They include the voltage pulse, the type of coil of the electromagnetic probe and the distance of the coil from the target. By manipulating those parameters we show how they influence the attack. Moreover parameters more specific to the target are also considered in this research. They include the x,y positioning of the coil over the target and the temporal settings of the attack. All of the parameters are described in Chapter 3 of this document. The ATMega 163+ smartcard and the LPC1769 micro-controller are tested against the attack, to show if the attack can produce the desired outcome. Those targets are described in Chapter 5. The targets execute proof of concept applications which can be found in the Appendices 8. The hardware and software setup of the environment is described in Section 4. The physics of the attack is described in Section 2. The tests performed on both targets, the conclusions and future research are respectively contained in Sections 6, 7, and 8.

1.2 Research Question

Is EMFI feasible on embedded systems or smartcards?

Testing shows if EMFI is successful in disrupting the designed operation of the embedded system and smartcard. Ideally, the goal is to force the system to change an instruction without halting the operation of the embedded system or smart card.

What is the most efficient configuration of the used EM probe?

All of the parameters tested in this research are relevant to this question. We answer it by describing how those parameters can influence the magnetic flux produced by the attacking device or how they can influence a specific target.

2 Electromagnetic Fault Injection

Electromagnetic Fault Injection is based on perturbing the target with a magnetic flux. A successful perturbation results in a computational fault of the device. More precisely certain instructions are influenced during the computation, resulting in an unexpected result. This section describes how the magnetic flux is caused and what influences it.

The electromagnetic probe generates a voltage pulse that sends current I through the coil. The current influences the magnetic field B around the coil. The magnetic flux Φ_B is the component of the magnetic field B passing through the surface of the coil. Figure 1 shows the magnetic field B of a coil, current I in the coil and surface S which is flat surface delimited by the coil.

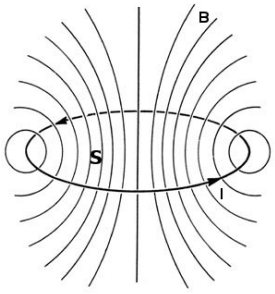

Figure 1: A magnetic field induced by current in the coil.

The following formula is used to calculate the magnetic flux, which is the component of the magnetic field passing through surface S:

$\Phi_B = BScos\Theta$, where:

B is the magnetic field
S is the area of the surface the magnetic flux is passing through
Θ is the angle between the normal of surface S and the magnetic field lines of B.

The formula for the magnetic field B of a one loop coil is:

$B = \frac{\mu_r I b^2}{2(b^2+z^2)^{\frac{3}{2}}}$, where:

μ_r is the permeability of the medium of the magnetic field
I is the current passing through the coil
b is the radius of the coil
z is the distance from the plane of the coil.

From those equations we can conclude that the measured magnetic flux is influenced by:

- the angle between the coil and the measurement device
- the distance of the coil from the measurement device

- the magnetic permeability of the material used for the coil's core
- the current running through the coil and the voltage pulse that generates it
- the size of the coil

Furthermore we can assume that:

- the angle between the measurement device and the magnetic flux should be 90° so that $cos\Theta = 1$, to measure a higher value of the magnetic flux
- a smaller distance between the coil and the measurement device will show a higher value of the magnetic flux
- a material with higher magnetic permeability will positively influence the magnetic flux
- a higher voltage pulse will generate more current and a stronger magnetic flux

The voltage pulse that generates the current has two characteristics: the amplitude of the pulse measured in Volts and the pulse duration measured in nanoseconds. Figure 2 shows the voltage of the perturbation received by the measuring device during the testing phase. The measured perturbation is 1.16V and the pulse duration is 20ns. The peak observed after the amplitude of the measured perturbation is the Inductive reactance of the coil[13]. It is caused by the sudden change of the current in the coil.

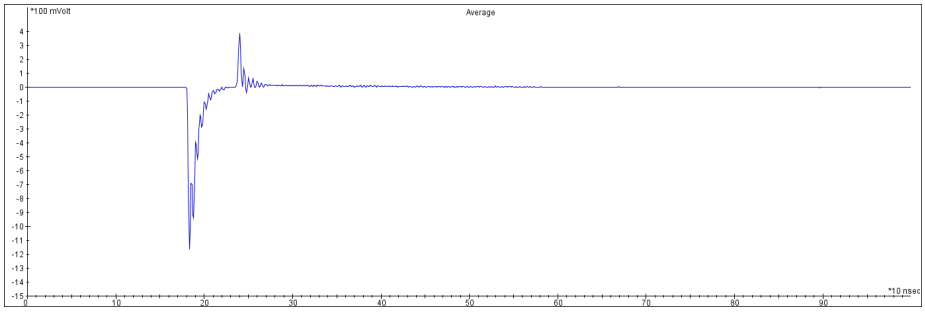

Figure 2: The voltage of the perturbation received by the measuring device.

3 Relevant Parameters

This chapter describes the selected parameters that were used during testing. The parameters determine the success of the attack. The tests presented in chapter 6 measure the voltage perturbation caused by the magnetic flux that is manipulated with those parameters. Target specific parameters which are described in Section 3.2 do not influence the magnetic flux. They define when the target is attacked and which part of the target should be attacked.

3.1 Magnetic flux parameters

The parameters that influence the magnetic flux directly and the voltage perturbation are:

- *Type of coil* that is used by the EM probe.
- *Type of core* of the coil. A ferrite and an air core were tested to establish the influence of the magnetic permeability.
- The *distance of the coil to the target* is the distance of the coil to the surface of the encapsulated chip not the chip itself.
- *Voltage supplied by the EM probe* by two capacitors and an external power supply . It is configurable between 60 and 90 Volts.
- *The angle between the coil and the target* was not included as a parameter in the tests in Chapter 6, because preliminary tests showed that 90 degrees yields the best results, which confirms the assumption from Chapter2.

The types of coils

All of the coils use one loop and have the following properties:

- a 1.5mm coil in diameter without a ferrite core.
- a 1.5mm coil in diameter with a cylindrical ferrite rod.
- a 1.7mm coil in diameter with an ep5 ferrite core.
- a 3mm coil in diameter with a cylindrical ferrite rod.
- a 3mm coil in diameter with an ep7 ferrite core.
- a 4mm coil in diameter with a cylindrical ferrite rod.
- a 5mm coil in diameter with a cylindrical ferrite rod.
- a 3mm coil in diameter with an ep7 ferrite core.

Distance between the coil and the target

Distance between the target and the coil is tested to prove the assumption made in Chapter2. As assumed a smaller distance should logically produce a higher voltage perturbation on the target. However, it is not determined what kind of minimum or maximum distance should be used to perform a successful fault injection on the targets. The distance was manually set with the provided table.

Voltage supplied to the EM probe

The voltage supplied by the EM Probe is 60V by default. Provided by a DC circuit with two electrolytic capacitors connected in a series. Additionally the external power supply is able to add up to 30V to the circuit. The external power supply allows us to vary the Voltage of the circuit from 60V to 90V in small intervals.

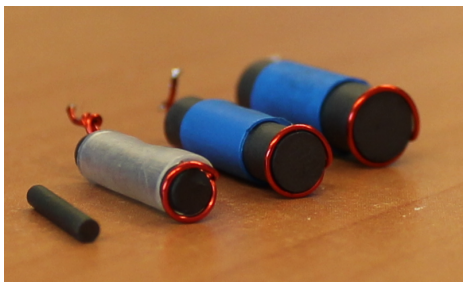

Figure 3: Coils with a cylindrical ferrite core used in this study. 5mm, 4mm and 3mm and 1.5mm in diameter.

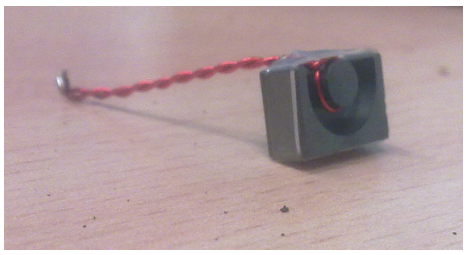

Figure 4: The EP7 ferrite core with a 3mm coil wrapped around it.

3.2 Target specific parameters

The following parameters are described as target specific, because their values are only dependant on the instruction flow on that target and the chips architecture. The settings of those parameters is only relative to a specific target. The parameters are mostly temporal except for:

- *Glitch cycles* - the number of times the EM probe generates a magnetic flux during one attack.
- *VCC/CLK Voltage* is the Voltage of the VCC line to the target device and the High part of the target's clock cycle.
- two dimensional position over the chip.

The temporal parameters include:

- *Wait cycles* which is the amount of cycles of the chip from the start of its operation. It is used to time the attack to target a specific instruction on the chip. It can be set to a specific cycle or changed with each attack within specified boundaries.
- *glitch length* which is the amount of time the EM Probe should continuously provide current through the coil. It can be set from 0 to 500, the maximum value being one clock cycle of the processor.
- *Glitch offset* which is the offset of the attack from the beginning of one clock cycle. It is set from 0 to 500, with 500 equal to one clock cycle of the processor. This value has to be recalculated for processors running at different frequencies.

All of the above are set in Inspector. Figure 5 illustrates those parameters.

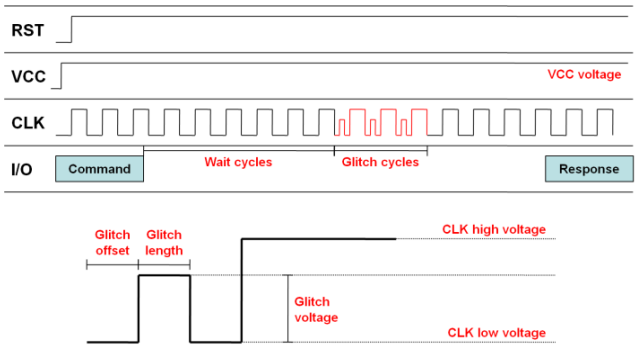

Figure 5: Inspector variables.[1]

4 EMFI Test environment

The following Chapter describes the hardware setup that was used during the tests. Figure 6 shows the setup of the devices and Figure 7 is the diagram of that setup. It shows how the devices are connected and how they interact with each other.

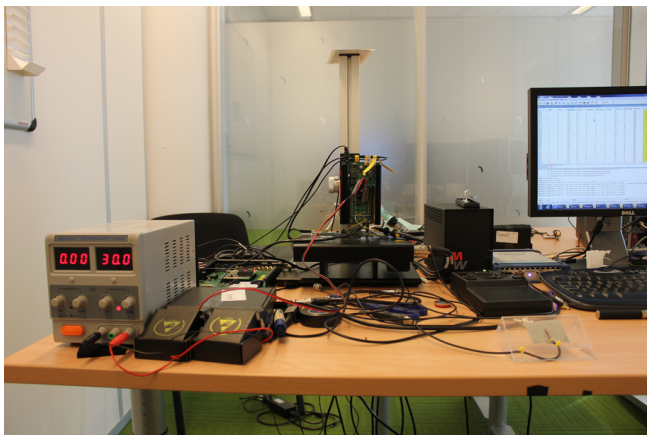

Figure 6: Hardware used for testing EMFI.

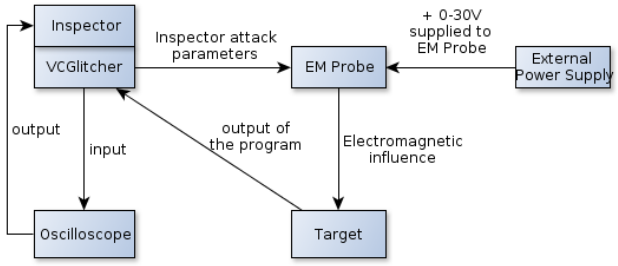

Figure 7: A diagram presenting the test environment.

Inspector is the software used to configure the attack and works with its hardware component VCGlitcher. It enables the communication with the oscilloscope (4.2), the targets (5) and the electromagnetic probe (4.5). The oscilloscope provides more feedback on the attack and is mainly used for verifying the attack. The electromagnetic probe receives the variables necessary to perform the attack from Inspector, through VCGlitcher. It is also connected to an additional power supply configured manually. The target is started with Inspector through VCGlitcher.
Figure 8 shows the following steps done during each attack:

1. Send attack parameters setup in Inspector.
2. Start the target.

3. Start the EM Probe with the received parameters.
4. Influence the device with the magnetic flux.
5. Send target output.
6. Pass target output.

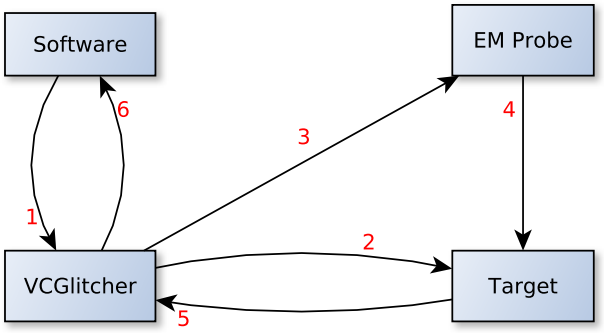

Figure 8: A simplified diagram presenting the steps of each attack.

4.1 The table

The EM probe is placed vertically so that the coil component is above the target. The coil can be manually moved closer and further away from the target. The table is movable in 2 dimensions and programmable in Inspector, which helps automatize the tests. With the table we can change the *positioning of the coil over the chip* and *the distance between the coil and the chip*, which are both configurable parameters of the attack. Figure 9 shows the table with the EM Probe and the embedded system.

4.2 The oscilloscope

The Pico 5203 oscilloscope[10] is used to measure the voltage perturbation by being connected to the measurement target with a BNC cable. It is also used to determine when a specific set of instructions is executed on the target. This is done by analyzing the power consumption of the target. Figure 10 shows the the power consumption of the ATMega 163 smartcard and the located pattern. Lastly, it is used to check if the targets and the EM Probe are operating as intended.

4.3 External power supply

The external power supply is directly connected to the EM Probe and can produce up to 30V of direct current. It is configured manually and has a coarse and fine tunning control. Figure 11 shows the power supply.

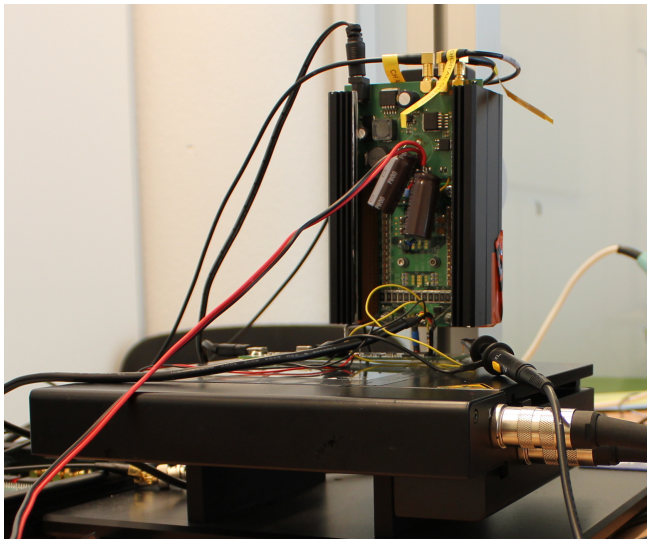

Figure 9: The table used to setup the target and the attacking device.

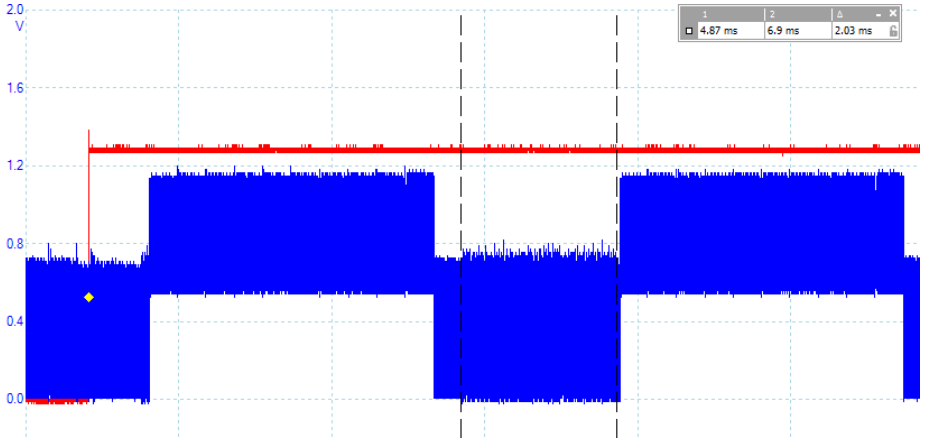

Figure 10: Blue signal: Oscilloscope readout of the power consumption of the ATMega 163. Red signal: An arbitrary trigger generated by VCGlitcher.

4.4 The measurement device

The measurement target is a one loop coil with a diameter of 0.5mm shielded by 0.7mm of plastic and connected to the oscilloscope with a BNC cable . It is used to measure the voltage perturbation generated by the magnetic flux. Figure 12 shows the device.

4.5 The electromagnetic probe

The prototype EM Probe developed by Riscure, uses a DC circuit with two electrolytic capacitors connected in series that are able to store 30V each, an external power supply and a single loop coil to generate the magnetic flux.

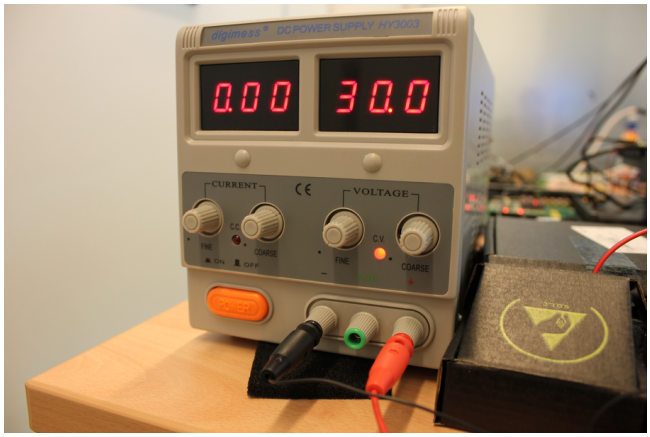

Figure 11: The external power supply.

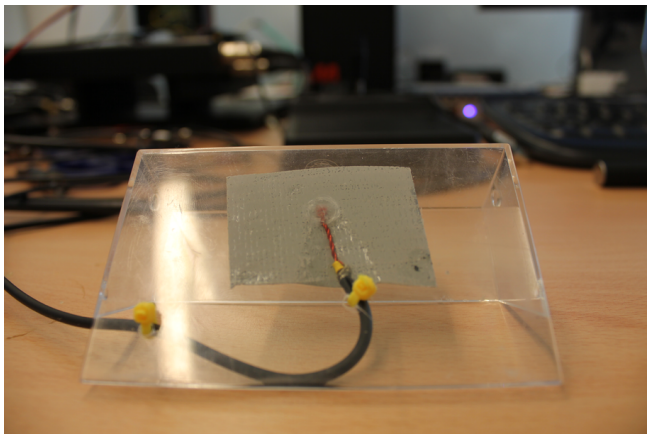

Figure 12: The measurement device.

The EM probe generates a voltage pulse that sends current through the coil to produce the magnetic flux in the magnetic field surrounding the coil in accordance with Maxwell's correction to Ampere's law[11]. The magnetic flux generates a perturbation in the voltage of the target device [2]. Such a voltage spike can damage or disable semiconductors[3] on the target. The coil has a ferrite core, which increases the inductance of the coil[4], in turn enhancing the magnetic flux, due to the core's permeability.

4.6 VCGlitcher

The VCGlitcher is a device developed by Riscure that is used for testing smartcards against different forms of fault injection. The device is controlled through Inspector and provides communication with the target device and the attacking device. It initializes the connected components with triggers configured in Inspector and collects the output. It is a workbench for fault injection. Figure 13 shows the device with an inserted smartcard.

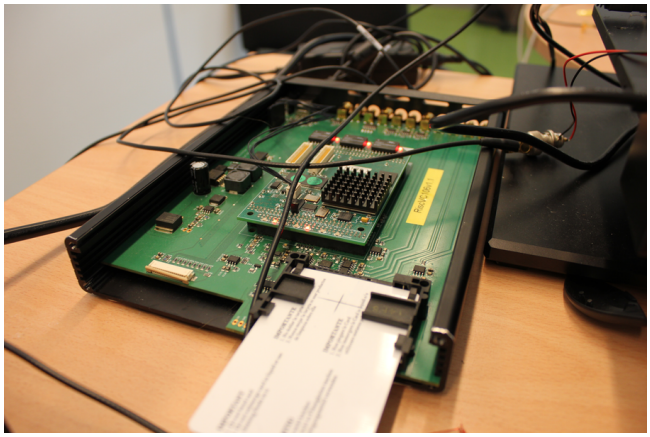

Figure 13: VCGlitcher with the top cover removed and an inserted smartcard.

4.7 Inspector

Inspector is a proprietary software developed by Riscure. In this study version 4.4 was used. Inspector is used to control the testing environment, namely VCGlitcher, the electromagnetic probe, the target and the movable platform on which the target is placed. Most importantly it is used to configure the parameters of the attack, trigger the attack and gather the output generated by the target as well as the output of the oscilloscope. The relevant part of Inspector for this study are the parameters the user is able to configure for the attack. The parameters described in Section 3.2 are set through the modules that are part of the software. Each type of attack is configured through its specific module. The modular architecture helps automate the experiments. Figure 14 shows the parameter configuration panel of the *Loop PoC perturbation* module developed for this research.

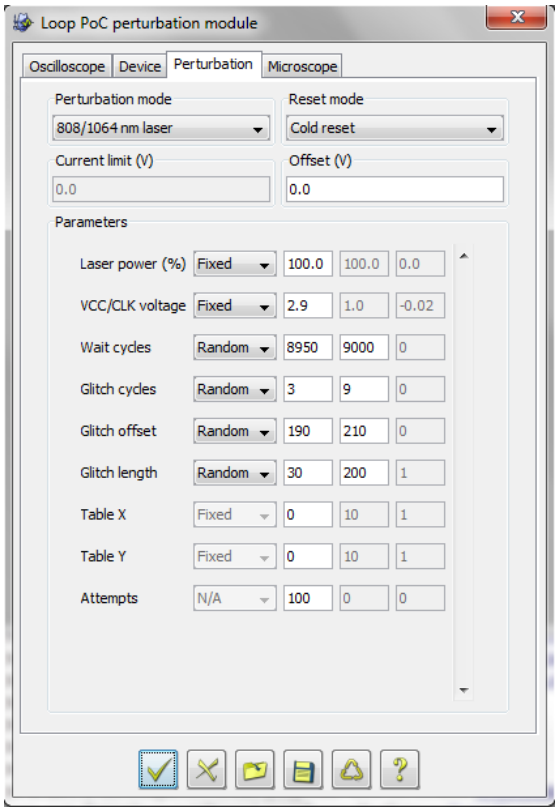

Figure 14: Parameter configuration in Inspector for the *Loop PoC perturbation* module.

5 EMFI Test Targets

5.1 ATMega 163

The ATMega 163+ is a smartcard with the following properties[6]:

- EEPROM: 512 bytes
- SRAM: 1024 bytes
- In-System Self-Programmable Flash: 16K bytes
- 8-bit CPU running at 1MHz

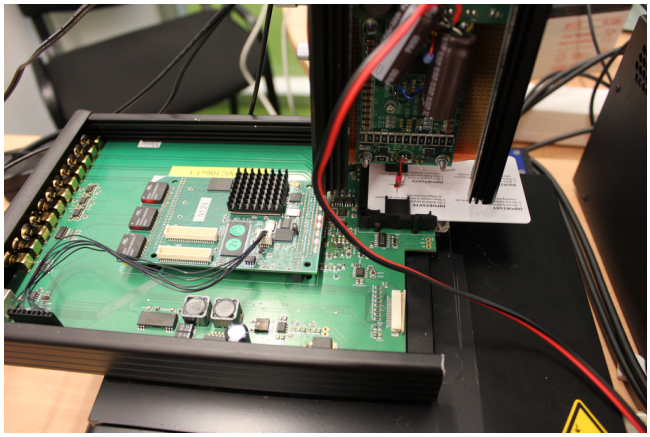

Figure 15: VCGlitcher with the top cover removed and an inserted smartcard.

Figure 15 shows the ATMega 163 smartcard inserted into VCGlitcher, with the EM probe targeting it from the front side of the encapsulated chip.

To determine if the target can be influenced with the electromagnetic probe a specific set of instructions was targeted. The target was programmed with instructions that were incrementing two variables. Determining if the attack was successful was done by checking the output of the variables returned by the smartcard. If they matched the intended values, the attack failed, yet if the value was different and the card still produced output in time, the attack was successful.

The target was prepared by compiling C code (Appendices 8) into assembly code and transferring that code onto the smartcard's flash memory. The C code executes 50 increments of two volatile integers set to 0. The variables need to be volatile so that the assembler performs the increments as separate instructions and not as one addition of 50 for each variable. Each of the additions takes 11 assembly instructions. This amount of operations provides a 2ms window to attack the device and influence the variables. Additionally before executing the aforementioned operations, two EEPROM writes are executed, to help identify when the execution of the increments start. Figure 16 shows the execution of the described instructions without the EEPROM writes. The increment instructions are selected in the figure by choosing the slightly noticeable pattern of the power

consumption of the target. This helps determine the time window of the attack. It also proves that the incremental instructions and not the write instructions of the variables are targeted.

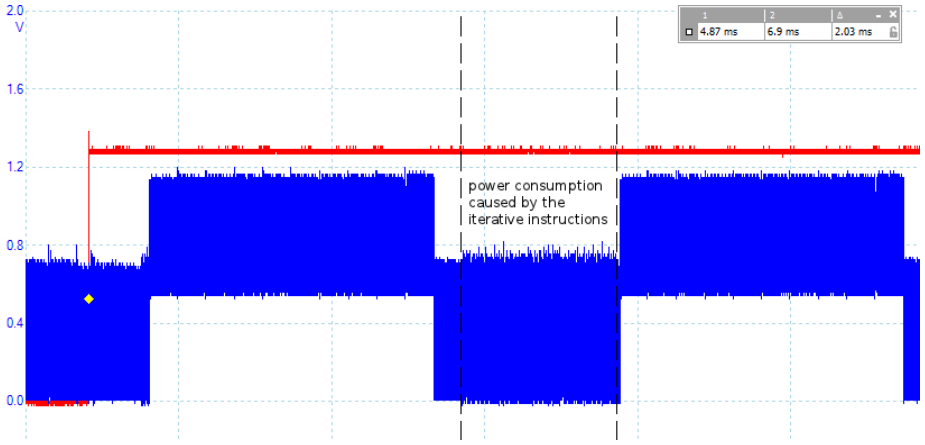

Figure 16: Blue signal: Oscilloscope readout of the power consumption of the ATMega 163. Red signal: An arbitrary trigger generated by VCGlitcher.

5.2 LPCXpresso 1769

The LPCXpresso 1769 consists of two components: the LPC-Link interface and an ARM based microcontroller, which is the target of this study.

The systems specifications include:

- flash memory: 512K bytes
- SRAM: 64K bytes
- 32-bit ARM Cortex-M3 running at 4MHz

This target was prepared with similar concepts to the smartcard. Although in this case, 200 iterations of two different volatile variables were used due to the faster processor. The C code that was compiled and stored on the device can be found in the Appendices 8

Timing the attack with the execution of the instructions was done by setting a trigger before the execution of the increments and releasing it afterwards to determine the timing of the operations. The system's trigger pin was connected to the oscilloscope to show when the instructions start and how much time they take. After the measurements the trigger was connected to the VCGlitcher, which instructs the EM probe to attack.

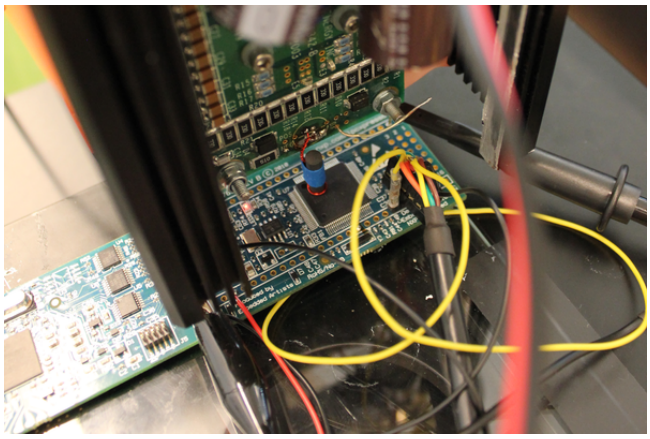

Figure 17: The LPCXpresso 1769 board.

6 Tests

6.1 Preliminary tests

The first set of tests is performed to establish if the provided setup can influence the execution flow of a target. For this purpose the following configuration is chosen:

- ATMega 163+ as the target that is assumed to be more prone to the attack due to less encapsulation.
- A 1.5mm coil with a ferrite rod.
- The VCC/CLK Voltage supplied to the target is set to 2.9V from the standard 5V. This is sound with the specification of the card and proven by the smartcard generating expected output when not attacked.[6]
- The distance is gradually set to minimal as a precautionary step. This ensures that the target does not get damaged.
- The voltage supplied to the EM probe is gradually increased from 60V to 90V, also as a precautionary measure to keep the target alive.
- *Glitch length*, *glitch offset*, *glitch cycles* and *position over the chip* are not constrained. Determining the correct values of those parameters is done by analysing previous attacks performed on the same target.
- The *wait cycles* are constrained to time the glitch with the targeted operations. This is done by analysing the power consumption of the chip with an oscilloscope.

The first set of tests was ran 400 000 times to help constrain the parameters set in Inspector for later tests. We determined that the ATMega's chip is sensitive to perturbation only in a specific position near its center. From this we can conclude that specific pins should be targeted during each consequent attack on a target.
After locating the vulnerable position the coil was locked in place and subsequent tests were performed without constraining *glitch length*, *glitch offset* or *glitch cycles*. The results of the attacks were grouped by the aforementioned parameters. Each of the parameters was constrained by a certain range if it did not produce any glitches in that range. The tests were repeated until approximately 20% of the attacks were successful and the parameters could no longer be easily constrained. We also observed that similar values of the *glitch offset* will produce an identical glitch. This is due to the perturbation affecting the same instruction. A specific number of unique faults is needed to obtain the cryptographic key through differential fault analysis. Therefore caution is advised when limiting the values of *glitch length*, *glitch offset* and *glitch cycles*.

6.2 Measuring the influence of distance

In the following set of experiments the distance of the coil from the target was changed to measure the voltage perturbation produced by the magnetic flux. The amplitude is registered with the measuring device connected to the oscilloscope,

described in Sections 4.4 and 4.2 respectively. The distance between the 1.5mm coil and the measuring device are set to 0.7mm, 1.2mm, 1.7mm, 2.7mm and 4.7mm. To ensure an accurate result an arithmetic mean is calculated from 10 runs of each setup.

The test is also repeated with different settings of the voltage pulse from the electromagnetic probe to measure the influence of distance for a magnetic fluxes of different force. The voltage was set from 60V to 90V in 5V increments. The following figure shows the amplitude of voltage perturbation caused by the magnetic flux from different distances.

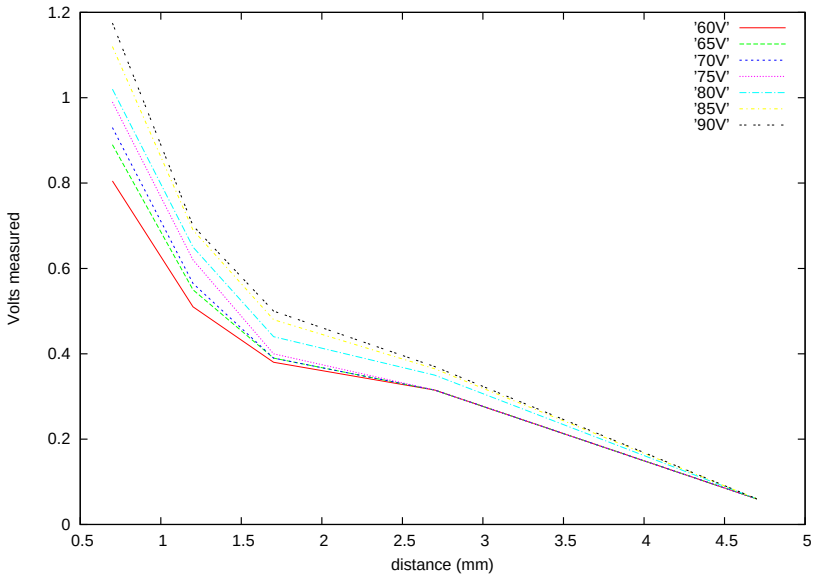

Figure 18: The influence of distance between the coil and the target on voltage perturbation.

At a shorter distance the amplitude is higher, which confirms the assumption from Chapter 2. The graph shows that distance has a significant influence on the amplitude of the voltage perturbation. Moreover we can observe that the loss of amplitude is exponential, for all of the tested voltage pulses.

6.3 Measuring the influence of the voltage pulse from the electromagnetic probe

To show the influence of the voltage pulse triggered by the electromagnetic probe the supplied voltage was changed starting at 60V up to 90V with 5V increments. A 1.5mm coil is used at the distances specified in Section 6.2. The voltage is manually changed on the external power supply connected to the electromag-

netic probe. The result of each setup is an arithmetic mean of 10 consequent tries.

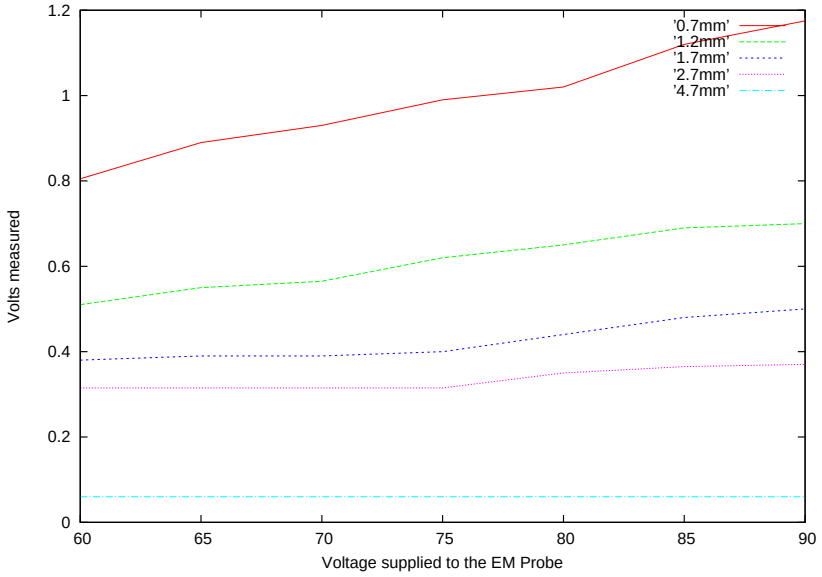

Figure 19: Influence of the electromagnetic probe's voltage pulse on the amplitude of voltage perturbation on the target.

Figure 19 shows a linear influence of the voltage pulse on the amplitude of voltage perturbation measured on the target. In the spectrum from 60V to 90V the effect is minor in comparison the the effect of distance. We can conclude that changing the voltage pulse is useful for fine tuning the magnetic flux generated by the coil.

6.4 Measuring the influence of different coils

All of the coils used in the following tests have a single loop and all except one are equipped with a ferrite core. The following coils are tested for the purpose of establishing which of the coils produces the highest amplitude of the voltage perturbation:

- round coil, 1.5mm in diameter, cylindrical ferrite core
- round coil, 1.5mm in diameter, no ferrite core
- round coil, 3mm in diameter, cylindrical ferrite core
- round coil, 4mm in diameter, cylindrical ferrite core
- round coil, 1.7mm in diameter, EP5 ferrite core

- round coil, 3mm in diameter, EP7 ferrite core

To show how the voltage perturbation is affected by each coil the voltage pulse was generated with 90V set on the electromagnetic probe. This increases the amplitude of the measured voltage perturbation and provides a larger spread of results with different coils. The tests are also repeated at different distances between the coil and the target. The distances are specified in Section 6.2. The results are an arithmetic mean of 10 consecutive measurements of each setting. Figure 20 shows the resulting perturbation caused by each coil at different distances.

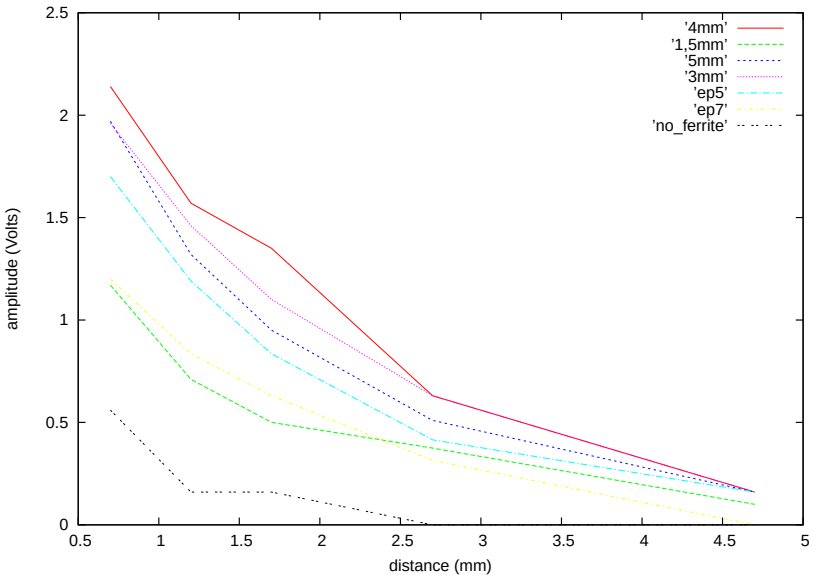

Figure 20: The amplitude of the voltage perturbation generated by different coils set at different distances from the target.

Figure 20 shows that the 4mm coil yields the best results producing an amplitude of 2.14V. The 3mm and 5mm coils both produce a smaller amplitude of almost 2V. The 1.5mm coil produces almost 1.2V. Moreover the influence of the ferrite core is shown when we compare the results of the 1.5mm coil with and without the ferrite core. This is due to the ferrite's [12] permeability that enhances the magnetic flux. The resulting amplitude of voltage perturbation on the target is raised from 0.65V without the ferrite core to 1.2V with the ferrite core.

A conclusion on which coil is most suitable for electromagnetic fault injection cannot be based solely on amplitude of voltage perturbation on the target. The area influenced on the target is also a factor, because coils of different sizes will

affect a different number of pins on the chip. Therefore the tests performed on the targets in Section 6.5 are performed with a 1.5mm and a 4mm coil. The 4mm coil is chosen because it produced the highest amplitude of voltage perturbation on the measurement device. The 1.5mm coil is chosen, because it's area of effect is much smaller. Moreover we test if a smaller amplitude of voltage perturbation affects the chosen targets.

6.5 Measuring the effectiveness of the EMFI attack on selected targets

The following tests show the effectiveness of the EMFI attack on the smartcard and the embedded system. The following setup was used:

- The 4mm and the 1.5mm coils are used.
- The voltage pulse of the electromagnetic probe is changed to fine tune the voltage perturbation on the target. This is done to observe any significant changes in the glitching rate caused by the perturbation. The tests start at 90V and continue until the targets do not produce a desirable amount of glitches.
- Each setup is repeated 1000 times to ensure a precise outcome.

Table 1 shows the statistics of the attacks on a smartcard. Table 2 shows the statistics of the attack on an embedded system.

Table 1: Statistics of the attack on a smartcard

Voltage pulse	Type of output	1.5mm coil	4mm coil
80V	glitched	0%	19%
	reset	0%	13%
85V	glitched	9%	21%
	reset	0%	15%
90V	glitched	20%	23%
	reset	0%	23%

Table 2: Attack ratio on an embedded system

Voltage pulse	Type of output	1.5mm coil	4mm coil
80V	glitched	0%	0%
	reset	0%	0%
85V	glitched	0%	0%
	reset	0%	0%
87.5V	glitched	0%	3%
	reset	0%	0%
90V	glitched	0%	8%
	reset	0%	0%

The 1.5mm coil tested on the embedded system did not produce any glitches. This is due to the thicker encapsulation of the chip, which influences the distance between the coil and the target. The 4mm coil caused a maximum glitch ratio of 9% with the maximal voltage pulse and at a minimal distance. Only 3 glitches were caused out of 1000 tries with the electromagnetic probe set to 87.5V and none at an 85V setting. Considering that the voltage pulse affects the magnetic flux in a linear way we can safely conclude that we are bordering on the lower amplitudes of successful perturbation for this target. We recommend to expose the target to a higher magnetic flux, to achieve a more feasible glitch rate.

The smartcard was glitched with both coils. The 4mm coil produced similar glitch rates of around 20% for all settings of the voltage pulse. This is not unexpected as it has a linear influence on the perturbation shown in Figure 19. The reset rate of the target reaches 23% and matches the glitch rate at the strongest settings of the attack. This confirms that the smartcard is more susceptible to the attack in comparison with the embedded system. Moreover we are bordering on the high amplitudes of successful perturbation for this target. The key on the smartcard has a chance to switch because of the reset rate, which was observed during preliminary tests. This is undesirable behavior as the smartcard becomes useless without its original key. A rapid drop in the reset rate and a smaller one in the glitch rate is observed voltage pulse is lowered. It also shows that the 4mm coil with the strongest settings is disruptive to the device. Moreover with a 1,5mm coil there are no resets. We also achieve a similar glitch rate with the 1,5mm coil at the strongest settings. The difference in glitch rate between both coils is 3% at the same voltage pulse. The same voltage pulse produces different amplitudes of voltage perturbation for both coils, as shown in Section ref:attdiffcoils. Considering the different amplitudes of the voltage perturbation for both coils and the similar glitch rate, we can conclude that the window for a successful attack ranges from 1.2V to 2.2V of the perturbation amplitude.

7 Conclusion

The research presented in this paper was performed to answer the following questions:

1. *Is EMFI feasible on embedded systems / smartcards?*
2. *What is the most efficient configuration of the used EM probe?*

The electromagnetic probe provided by Riscure was used to introduce faults in an ATMega163+ smartcard and the LPC1769 micro-controller. Both targets were running proof of concept code that iterated two volatile variables over a short period of time. Both of them returned unexpected output while under the influence of the magnetic flux generated by the probe proving that EMFI is feasible on both systems.

In Chapter 6 we showed how the configuration of the electromagnetic probe, through specific parameters influences the magnetic flux, which was measured as an amplitude of voltage perturbation on the target. Those measurements were supported by specific formulas used to calculate the magnetic flux at a given point presented in Chapter 2. From those formulas assumptions were formed about the influence of the voltage pulse and the distance from the coil. We proved the assumptions stating that the smallest distance and the highest voltage pulse produce the highest amplitude of voltage perturbation. We also showed how that perturbation affects different targets, by using a 1.5mm and a 4mm coil on the electromagnetic probe. As was expected the type of coil influences not only the generated perturbation but also the target in a more complex way. Comparing the results of attacks on the targets shown in Tables 1 and 2, we can clearly see that neither of the supplied coils is superior over the other in a universal way. Either because of the power of the magnetic flux they produce, or the area they can affect.

To answer the second research question; the most efficient configuration of the used electromagnetic probe on the smartcard target is the voltage pulse set to 90V, the distance from the target set to minimum and a 1,5mm coil with a ferrite core. the most efficient configuration when targeting the embedded device is the voltage pulse set to 90V, distance set to minimum and a 4mm coil with a ferrite core. Moreover as shown in Section 6.5 a higher amplitude of voltage perturbation does not always work best depending on the target.

8 Future Research

The EP shape of the ferrite core results were inconclusive and it would be useful to repeat tests with different sizes of coils of the same shape to possibly find a pattern and establish when such a ferrite core shape is beneficial.

It would also be interesting to provide more power to the EM Probe to see what amount of amplitude is harmful for each of the devices. Moreover testing the more precise 1.5mm coil on the embedded system using a higher voltage pulse could present interesting results.

Acknowledgements

I would like to thank Riscure for providing the hardware and software necessary to carry out this research and Fred de Beer for providing the prototype of the electromagnetic probe. Also Niek Timmers provided very valuable input during the discussions we had during the project.

References

[1] **Inspector 4.4 Manual**
Riscure
As seen on: January 2012

[2] **Wikipedia: Electromagnetic pulse**
http://en.wikipedia.org/wiki/Electromagnetic_pulse
As seen on: 12 February 2012

[3] **Wikipedia: Voltage spike**
http://en.wikipedia.org/wiki/Voltage_spike
As seen on: 12 February 2012

[4] **Wikipedia: Ferromagnetic core inductor** http://en.wikipedia.org/wiki/Inductor#Ferromagnetic_core_inductor
As seen on 12 February 2012

[5] **NXP's LPCXpresso website**
http://ics.nxp.com/lpcxpresso/~LPC1769/
As seen on 14 February 2012

[6] **Atmel's ATMega 163 Documentation**
http://www.atmel.com/Images/doc1142.pdf
Revision E, updated: February, 2003

[7] **NXP's LPC1769 datasheet**
http://ics.nxp.com/products/lpc1000/datasheet/lpc1763.lpc1764.lpc1765.lpc1766.lpc1767.lpc1768.lpc1769.pdf
Updated: November 14, 2011

[8] **Wikipedia: Brute-force attack**
http://en.wikipedia.org/wiki/Brute-force_attack#Theoretical_limits As seen on: 11 February 2012

[9] **Wikipedia: Differential Fault Analysis**
http://en.wikipedia.org/wiki/Differential_fault_analysis
As seen on: 11 February 2012

[10] **Picoscope 5000 Series documentation**
www.picotech.com/document/pdf/ps5000-en-7.pdf
As seen on: 11 February 2012

[11] **Wikipedia: Maxwell's equations**
`http://en.wikipedia.org/wiki/Maxwell's_equations`
As seen on: 11 February 2012

[12] **Wikipedia: Permeability**
`http://en.wikipedia.org/wiki/Permeability_%28electromagnetism%29`
As seen on: 11 February 2012

[13] **Wikipedia: Inductive reactance**
`http://en.wikipedia.org/wiki/Electrical_reactance#Inductive_reactance`
As seen on: 6 May 2012

Appendices

The source code in C for ATMega

main.c

```
#include <avr/io.h>
#include <util/delay_basic.h>
#include <avr/eeprom.h>
#include <avr/interrupt.h>
#include "definitions.h"
#include "basic_io.h"
#include "crypto.h"
#include "main.h"
#include "utils.h"

/* sebastian */
#include "loop.h"

#include <stdlib.h>

#define MAX_PIN_CTR 3


/** Global vars **/

uint8_t buffer[35];

/** Global vars define elsewhere **/
extern unsigned char deskey[8];


//extern EEMEM variables

extern unsigned char EEMEM ee_deskey[8];

/** Local functions **/

void do_des();
void set_key();
void set_pin();
inline void des_decrypt();
void verify_pin_single();
void verify_pin_double();
void better_pin_double();
void reset_pin();
void is_auth();



unsigned char pin[4]={8,2,6,9};
```

```
unsigned char EEMEM ee_pin[4]= {8,2,6,9};
int EEMEM ee_seed[sizeof(int)];

uint8_t pin_ctr = MAX_PIN_CTR;
uint8_t auth=FALSE;
uint8_t EEMEM ee_pin_ctr = MAX_PIN_CTR;

void process(){


switch(buffer[CLA]){
case 0x00: // select application, or whatever starts with CLA=0x00
respond(0x08);
break;
case 0x77:
iterations();
break;
.
.
.
.
.
break;
default: // illegal card usage
respond_code(0x00, SW_UNKNOWN_msb, SW_UNKNOWN_lsb);
break;
}

}

int main(){
int seed;
//Retrieve EEPROM data
eeprom_read_block(deskey,ee_deskey,16);
eeprom_read_block(pin,ee_pin,4);
eeprom_read_block(&seed,ee_seed,sizeof(int));
pin_ctr=eeprom_read_byte(&ee_pin_ctr);
srand(seed);
//Enable global interrupts
sei();

//And proceed
initialize();
sendATR();

while(1){
//Read Command
readAPDU();
//Check command and act accordingly
determine();
```

```
}
//unreachable code... just avoiding compiler complaints
return 0;
}
```

loop.c

```
#include <stdint.h>
#include <stdio.h>
#include <avr/eeprom.h>

void bit32Out(uint32_t inter) {
uint8_t A = inter >> 24;
uint8_t B = inter >> 16;
uint8_t C = inter >> 8;
uint8_t D = inter >> 0;
byteOut(A);
byteOut(B);
byteOut(C);
byteOut(D);
}

int EEMEM ee_flag;

void iterations() {
volatile uint32_t i = 0;
volatile uint32_t j = 0;
volatile uint8_t k = 0;

byteOut(k);

eeprom_write_byte(&ee_flag,0xAA);
eeprom_write_byte(&ee_flag,0xAA);

i++;
j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;

eeprom_write_byte(&ee_flag,0xAA);
eeprom_write_byte(&ee_flag,0xAA);

bit32Out(i);
bit32Out(j);
}
```

The source code in C for LPC 1769

main.cggggg

```
/*
===============================================================================
 Name        : main.c
 Author      :
 Version     :
 Copyright   : Copyright (C)
 Description : main definition
===============================================================================
*/

#ifdef __USE_CMSIS
#include "LPC17xx.h"
#endif
#include <stdio.h>
#include "uart0.h"
#include <string.h>

#define PIN_PULLUP      0UL
#define PIN_REPEAT      1UL
#define PIN_NONE        2UL
#define PIN_PULLDOWN    3UL

#define TRIG ((uint32_t)12)
#define TRIG_SEL_MASK    ~(3UL << 24)
#define TRIG_SET_MASK    (1UL << 12)
#define TRIG_CLR_MASK    ~(TRIG_SET_MASK)
#define TRIG_AS_OUTPUT   LPC_PINCON->PINSEL4&=TRIG_SEL_MASK;LPC_GPIO2->FIODIR|=TRIG_SET_MA
#define TRIG_AS_INPUT    LPC_GPIO2->FIOMASK &= TRIG_CLR_MASK;
#define TRIG_SET         LPC_GPIO2->FIOSET = TRIG_SET_MASK
#define TRIG_CLR         LPC_GPIO2->FIOCLR = TRIG_SET_MASK
#define TRIG_IS_SET      ((LPC_GPIO2->FIOPIN & TRIG_SET_MASK)?1:0)
#define TRIG_IS_CLR      !(TRIG_IS_SET)
#define TRIG_TOGGLE      TRIG_IS_SET?TRIG_CLR:TRIG_SET
#define TRIG_MODE(x)     LPC_PINCON->PINMODE4&=TRIG_SEL_MASK;LPC_PINCON->PINMODE4|=((x&0x3
#define SETTRIG TRIG_SET
#define CLRTRIG TRIG_CLR

int main(void) {
TRIG_AS_OUTPUT;
TRIG_MODE(PIN_NONE);

UART0_Init(115200);




while(1) {
```

```
uint8_t start = 0;
uint8_t charec = 0;

while(start != 0x41){
scanf("%c", &start);
//printf("%x\n", start);
}

charec = getchar();
if(charec == 0x42) {
volatile int j = 0;
volatile int i = 0;
SETTRIG;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;i++;j++;
CLRTRIG;
printf("%x",i);
printf("%x",j);
}
else if(charec == 0x43) {
printf("OK: c\r\n");
}
else {
printf("unknown");
}
}


printf(" ** Finished - entering infinite loop **\r\n");
// Enter an infinite loop, just incrementing a counter
volatile static int i = 0 ;
```

```
while(1) {
i++ ;
}

return 0 ;
}
```