IST-2002-507932

ECRYPT

European Network of Excellence in Cryptology

Network of Excellence

Information Society Technologies

D.VAM.5
Report on DPA and EMA attacks on FPGAs

Due date of deliverable: 31 July 2005
Actual submission date: 31 July 2005

Start date of project: 1 February 2004 Duration: 4 years

Lead contractor: UCL Crypto Group

Revision 1

Project co-funded by the European Commission within the 6th Framework Programme		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission services)	
RE	Restricted to a group specified by the consortium (including the Commission services)	
CO	Confidential, only for members of the consortium (including the Commission services)	

Report on DPA and EMA attacks on FPGAs

Editor
François-Xavier Standaert (UCL)

Contributors
Lejla Batina (KUL), Elke De Mulder (KUL), Kerstin Lemke (RUB),
Nele Mentens (KUL), Elisabeth Oswald (IAIK), Eric Peeters (UCL).

31 July 2005
Revision 1

The work described in this report has in part been supported by the Commission of the European Communities through the IST program under contract IST-2002-507932. The information in this document is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

Contents

Chapter 1

Introduction

Since their publication in 1998, power-analysis and electromagnetic (EM) attacks have attracted significant attention within the cryptographic community. So far, they have been successfully applied to different kinds of (unprotected) implementations of cryptographic algorithms. Most of the attacks published in the open literature apply to software implementations for smart cards (see [21], [28] or [29]). Recently however, the analysis of hardware implementations and in particular on Field Programmable Gate Arrays (FPGAs) has become increasingly popular.

As part of a modern design flow, FPGAs are gaining more importance. Reasons for this include their relatively low cost and the available tools. High-level descriptions (like VHDL for example) of circuits can easily be ported and directly used, for testing or real application purposes. In addition, FPGAs are highly attractive solutions for hardware implementations of encryption algorithms and numerous papers underline their growing performances and flexibility for any digital processing application.

Regarding these potential uses of FPGAs in secure applications, it is natural to investigate whether these platforms may be reliable with respect to the various physical attacks that can be considered against microelectronic devices. A general study of these security issues has been conducted in [55] and suggested different potential security threats. Among the various open questions pointed out in this analysis, the possibility to carry out side-channel attacks against reconfigurable hardware devices was of particular of interest, since these attacks had already been shown particularly efficient against small devices such as smart cards. It gave rise to a number of public investigations.

As a consequence of these investigations, this report describes the various realizations of power-analysis and EM attacks on FPGAs. All setups and attacks that we describe have been performed by members of the VAMPIRE lab. We show that FPGAs leak a significant amount of information about internal computations through the supply lines and through EM channels. Then, we consider two particular case studies: elliptic-curve point-multiplication and block ciphers. Through a number of practical attacks, we illustrate that FPGA implementations of these cryptosystems can be defeated, in a similar way as smart card implementations. A number of commercial devices are used for the evaluation of these attacks, although the principles discussed are likely to be applicable to any similar reconfigurable hardware device.

The rest of this report is organized as follows. We review some basic facts about FPGAs in Section 2. In Section 3, we survey the setups that have been built by members of the VAMPIRE lab. In Section 4, we report on power analysis attacks against implementations of block ciphers. In Section 5 we report on power-analysis and EM attacks on implementations of elliptic curve cryptosystems. We conclude this report in Section 6.

Chapter 2

FPGA Technology

In the design of embedded systems, Application Specific Integrated Circuits (ASICs) have traditionally been common components for providing the high performance and/or low power designs that many systems require at the expense of long and difficult design cycles. In the 1980s the use of reprogrammable components, in particular FPGAs, was introduced. FPGAs allow faster design cycles because they enabled early functionality testing. Nonetheless, the performance and size of FPGAs did not permit them to substitute ASICs in most applications and thus, they were mainly used to prototype embedded chips small enough to fit in the FPGA. In recent years, however, FPGA manufacturers have come closer to filling the performance gap between FPGAs and ASICs, enabling them, not only to serve as fast prototyping tools but also to become active players as components in embedded systems.

The trend in both industry and academia is to develop chips that are constituted with embedded components such as memory, I/O controllers, and multiplier blocks as well as (more recently) programmable cores. The resulting integrated systems are known by various names ranging from hybrid architectures to Systems-on-Chip (SoC), Configurable System-on-Chip (CSoC), Reconfigurable Systems-on-Chip (RSoC), and Systems on Programmable Chip (SoPC), among others. Thus, reconfigurable devices and in particular FPGAs are usual parts of present embedded systems. This fact is exemplified by the great number of research publications in the area of FPGAs and applications such as image processing, computer vision, solution of pattern recognition problems, e.g., text searching, fingerprinting matching, etc., solution of boolean satisfiability problems, digital signal processing, and many others.

The reconfigurability of FPGAs offers major advantages when using them for cryptographic applications. Despite the vastness of the research literature on FPGA cryptographic implementations, there is only a few work regarding the suitability of FPGAs for security applications from a system point of view. In particular, very little work has been done on the resistance of FPGAs to physical or system attacks.

2.1 Types of FPGAs

Two main classes of FPGA architectures can be distinguished. Coarse-grained architectures consist of fairly large logic blocks, often containing two or more look-up tables and two or more flip-flops. Fine-grained architectures consist of a large number of relatively simple logic blocks. Another difference in the architectures is the underlying process technology used to manufacture the device. Currently, the highest-density FPGAs are built using static memory (SRAM) technology, which is similar to microprocessors. The other common process technology is called anti-fuse, which features better programmable interconnections.

SRAM-based devices are inherently re-programmable, even in-system. After a power-up is applied to the circuit, the program data defining the logic configuration must be loaded in the SRAM [25]. The program data defines how each of the logic blocks functions, which I/O blocks are inputs and outputs, and how the blocks are interconnected. The FPGA either self-loads its configuration memory, or an external processor downloads the memory into the FPGA. The configuration time is typically less than 200 ms, depending on the device size and configuration method. In contrast, anti-fuse devices are one-time programmable (OTP). Once programmed, they cannot be modified, but they also retain their program when the power is off. Anti-fuse devices are programmed in a device programmer either by the end user or by the factory or distributor.

2.2 The Xilinx Virtex Architecture

Virtex devices feature a flexible, regular architecture that comprises an array of configurable logic blocks (CLBs) surrounded by programmable input/output blocks (IOBs), all interconnected by a rich hierarchy of fast, versatile routing resources. Virtex FPGAs have a coarse-grained architecture, are SRAM-based, and are customized by loading configuration data into internal memory cells. The basic building block of the Virtex CLB is the logic cell [59]. A logic cell includes a 4-input function generator, carry logic, and a storage element. The output from the function generator in each logic cell drives both the CLB output and the D input of the flip-flop. Each Virtex CLB contains four logic cells, organized in two similar slices. Figure 2.1 shows a more detailed view of a single slice. In addition to the four basic logic cells, the Virtex CLB contains logic that combines function generators to provide functions of five or six inputs.

The Virtex function generators are implemented as 4-input look-up tables (LUTs). In addition to operating as a function generator, each LUT can provide a 16×1-bit synchronous RAM. The Virtex I/O Block features SelectIO inputs and outputs that support a wide variety of I/O signaling standards [59]. Some of the possible I/O standards require VCCO (output supply) and/or VREF (reference) voltages. These voltages are connected to the device pins that serve groups of IOBs, called banks. Consequently, not all I/O standards can be combined within a given bank. Each bank has multiple VCCO pins, all of which must be connected to the same voltage. This voltage is determined by the output standards in use.

2.3 Configuration of the FPGA

The configuration data stream determines the functionality of the CLBs of which an FPGA is composed of (see Figure 2.2). CLBs are connected by programmable interconnections and are arranged in an array structure such that programmable interconnections can be realized

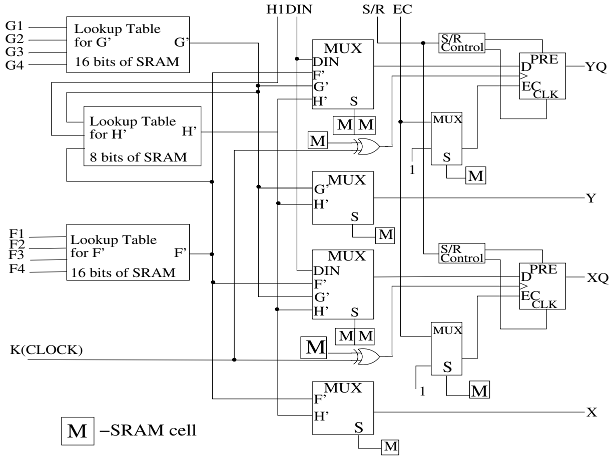


Figure 2.1: Simplified diagram of a single Xilinx Virtex slice

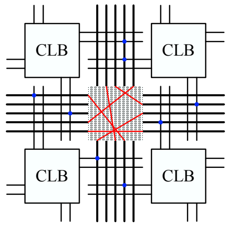


Figure 2.2: Configurable CLBs and configurable interconnects

by switches. Large FPGAs—like the Virtex series from Xilinx which contains several hundred CLBs—offer in addition configurable memory blocks, which can implement RAMs and ROMs efficiently. Modern FPGAs allow implementing digital circuits with a complexity of multi-million system gates and they get along with clock frequencies of 100 MHz and beyond. In the following, we will only consider the security of FPGAs once they have been configured, although the configuration process itself may be the subject of security threats as well.

Chapter 3

FPGA Measurement Setups

Power-analysis and EM attacks are rather powerful types of side-channel attacks. This is because the instantaneous power consumption of a typical device is usually closely related to the data and the instruction that is executed on a certain moment in time. Since the power consumption results from small currents flowing inside electronic devices, the EM field radiated by a device is similarly data-dependent. In the subsequent sections we sketch measurement setups for FPGAs. They generally apply to both power and EM measurements.

3.1 Power Consumption Characteristics of FPGAs

Nowadays, almost all devices are implemented in CMOS (Complementary Metal-Oxid Semiconductor) technology. In CMOS technology, the values 0 and 1 are represented by GND and VCC, respectively. The dominating factor for the power consumption of a CMOS gate is the dynamic power consumption [53]. *Transition count* leakage and *Hamming weight* leakage can typically be observed in CMOS circuits, see [28] for a detailed explanation.

The power consumption behavior of a CMOS processor can be roughly sketched as follows. On every rising edge of the clock, the simultaneous switchings of the gates cause a current flow which is observable through both the GND and VCC pins of the device. This current flow can be observed on the outside of the device by (for example) putting a small resistor (or a current probe) between the devices GND pin (or VCC pin) and the external GND (or VCC). The current flowing through the resistor creates a corresponding voltage signal which can be measured by a digital oscilloscope.

The characterization of the power-consumption of FPGAs has received little attention so far. Relatively recently, Shang *et al.* presented results in that field [39]. In their article, they analyze the dynamic power consumption of the XILINX Virtex-II family. They conclude that 60% of the dynamic power consumption is due to the interconnects, 14% is due to the clocking, 16% is due to the logic and 10% is due to the IOBs.

Because of the relatively complex structure and ability to perform parallel computing of FPGAs, it was initially assumed that conducting side-channel attacks would be hardly possible against these devices. However, in spite of these particular features, such attacks can be realized in practice.

The state-of-the art attacks usually only assume a very simple power consumption model, based on the evaluation of the number of bit transitions inside a circuit, e.g. the Hamming distance power consumption model, mentioned in a number of references, e.g. [28]. However, more accurate characterizations of the leakages, e.g. inspired by Shang et al. in [39], would probably allow to improve the efficiency of these attacks. As already mentioned, side-channel attacks against FPGAs are a relatively recently investigated topic and would deserve further research.

3.2 Evaluation of an existing FPGA-Board

Using existing FPGA boards to conduct power-analysis attacks is a natural choice for preliminary investigations. This approach was selected at IAIK, using the Xess XSV800 prototyping boards [56]. The FPGA of the XSV800 prototyping board is a Xilinx XCV800 [57], which has a core voltage of 2.5 V. Normally this voltage is supplied by an on-board regulator, but the XSV800 also allows an external source to supply the core voltage. This allows measurements of the FPGA core's power consumption.

To determine if measurements with a sufficiently high signal to noise ratio can be made, power consumption measurements were conducted in practice. A reference circuit was designed in VHDL with the purpose to cause significant peaks in the FPGA's power consumption trace; the design consisted of approximately 6,000 flip-flops which were alternatingly loaded with all ones and all zeros. The CLBs of an FPGA consist, amongst other circuitry, of ordinary CMOS flip-flops, as described in Section 2. Therefore, the power consumption characteristics of an FPGA should roughly correspond to those of an ordinary CMOS circuit (except the influence of the interconnects).

The reference design also provided a trigger impulse shortly before the flip-flops were loaded with all ones. This signal was used to trigger the start of a measurement with a digital oscilloscope. The core supply voltage of the FPGA was supplied externally by a laboratory power supply unit. A resistor (shunt) was put in series with the supply voltage line. Then, the FPGA was configured with the reference design, and the power consumption of the core was measured over the resistor with the digital oscilloscope.

Conceptually, the power trace should have displayed a peak value at the moment where the flip-flops were loaded with all ones. Although a peak could be determined in the trace (with extreme zooming), it was superposed by noise signals of magnitudes several decades higher. The source of this noise can be found in the several additional chips of the XSV800 board (e.g. SRAM chips). Moreover, the fast-switching operations draw current from the bypass capacitances of the FPGA, and only their recharge current can be measured. The course of this recharge current is much flatter than the FPGA's power consumption and therefore the measured peaks are also very low.

Because of these results, the utilization of the XSV800 prototyping board was disapproved. Instead, it was decided to build a custom FPGA-board specifically designed for the purpose of power analysis attacks. A similar approach was adopted by two other members of the VAMPIRE lab. We describe these boards in the following sections.

Remark that the quality of the measurements obtained with a specific custom board highly depends on the suppression of the noise of the other components of the board, either by bypassing or removal. On the other hand, this also yields a context that is a less generic than targeting a general purpose board and therefore means a more expensive attack context. Stabilizing capacitors may also be removed from the boards of recent devices. Due to the high working frequencies of recent devices, these capacitors are required to provide a regular power supply to the chips. A side-effect of these capacitors is that they filter the power consumption measurements. Removing them can consequently be interesting for measurement purposes, as long as the adversary has the ability to control the clock frequency of the chip, such that the circuit still behaves properly. In the following descriptions, different approaches were chosen by the ECRYPT partners: from still relatively generic boards (e.g. COSIC, IAIK) to very rudimentary boards containing the target FPGA only (e.g. UCL).

3.3 The FPGA Measurement Board of COSIC

This setup consists of two boards (see Figure 3.1). The mother board is responsible for interfacing the PC via the parallel port. It can be connected with the XILINX parallel cable in order to program the FPGA (and the configuration PROMs) and it provides some LEDs, switches and buttons for testing purposes. The daughter board itself just carries the FPGA, it allows to access some pins for triggering and to measure the power consumption of the FPGA in a convenient way.

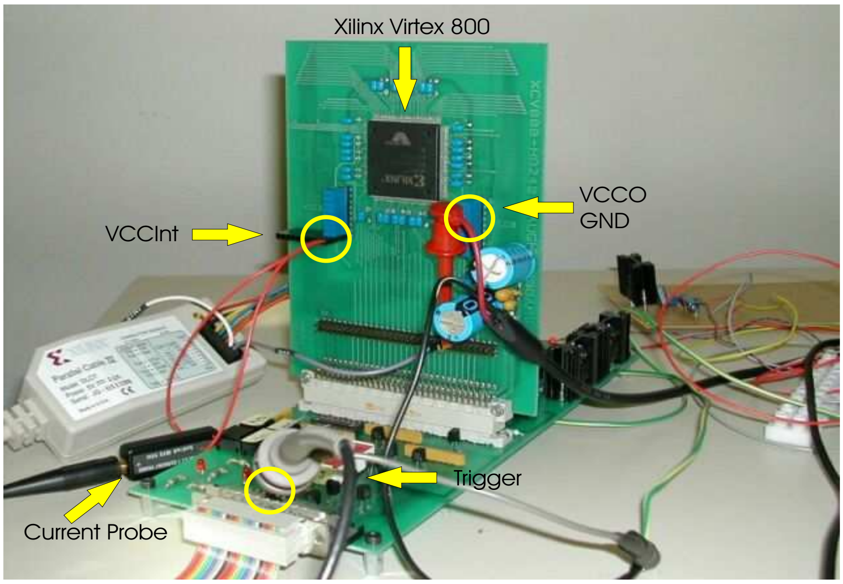


Figure 3.1: The COSIC measurement setup. On the daughter board the current probe is connected to VCCINT. Alternatively it can be connected to the VCCO of the individual banks, or the GND.

A protocol was designed to send and receive data to and from the FPGA. When the FPGA communicates with the PC, it uses the three most significant bits of the status lines to indicate its status. The two remaining bits of the status lines are used for sending the result from the FPGA to the PC. The PC uses the control lines to send the commands and it uses the data lines to send the input data to the FPGA. The protocol is independent from the operations executed in the FPGA. The length of the communicated data is controlled by the PC. This provides a flexible setup where experiments with different algorithms can be performed in a coherent manner.

A Xilinx XCV800 FPGA from the Virtex series in a HQ240C package is used in this setup. Reasons for this particular choice include:

1. The resources are sufficient to implement a fully parallel 160-bit elliptic-curve point-multiplication.

2. This is the most powerful FPGA that can be used for hand-mounting on the board. This is because the pins of this FPGA are on its sides. The more powerful FPGAs have the pins underneath with a grid structure, and so special machines are needed to mount them.

3. The architecture is made of combinational and memory elements. Because of this property it is a good representative of application specific integrated circuits (ASICs).

The XCV800 has 12 core voltage supply (VCCINT) pins, 16 output voltage supply (VCCO) pins and 32 ground (GND) pins. The FPGA is divided into 8 banks each with their own VCCINT and VCCO pins. After the implementation of the desired circuit and the configuration of the FPGA with the implementation data, some banks will be used more frequently than others; these banks draw more current from their supply lines. In case that different parts of a design are mapped to different banks of the FPGA, measuring the current of the individual banks allows to take more precise measurements for them. By measuring VCCINT and VCCO of the same bank separately, we can detect the input/output and core activity timing and power consumption separately.

Therefore we use three headers with two lines for VCCINT, VCCO and GND as shown in Figure 3.1. During the normal operation of the board without measurement the two pins are connected by a jumper. When it is desired to measure the current flow from a specific bank, the associated jumper is replaced by a cable that is going through the hole in the current probe as shown in Figure 3.1.

There are no oscillators present on the boards. Hence, clock signals must be provided by an external pattern generator. $10nF$ capacitors are placed between every VCCINT and VCCO pin of the FPGA and the nearest GND.

With this setup power-analysis as well as EM attacks have been performed. Additional information can be found in [6] .

3.4 The FPGA Measurement Board of IAIK

At the core of the board is a Xilinx XCV300E FPGA of the Virtex-E series. The power supply and ground pins of the FPGA are connected to the power and ground lines over measurement pins. This setup allows easy measurement of the current on a specific power line by connecting its pins over a sensor resistor. The pins of lines which are not used for measurement must be connected with a jumper.

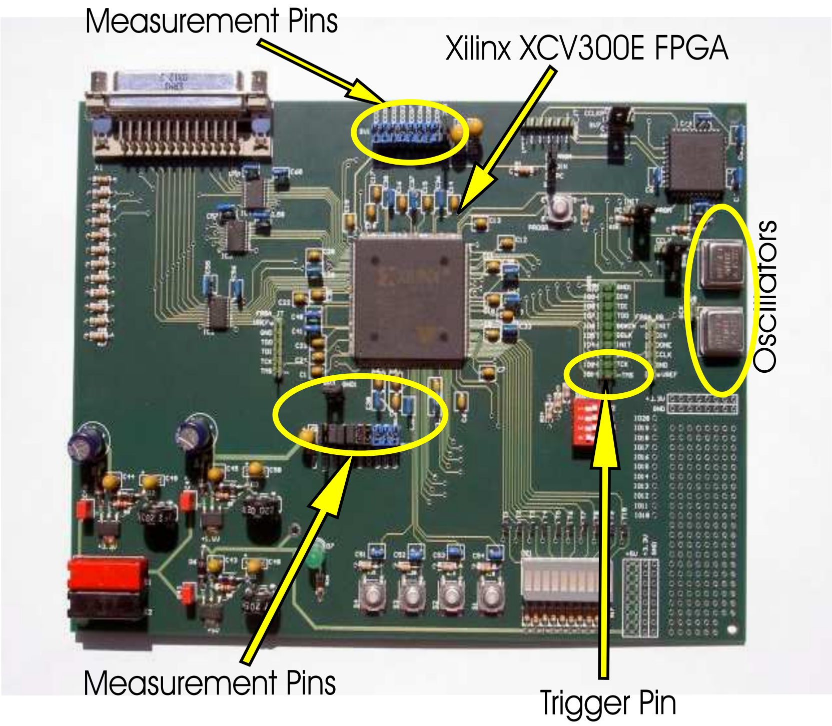


Figure 3.2: The IAIK measurement setup.

The board offers different configuration options for the FPGA including configuration from the PC. Non-volatile configuration is possible with an on-board Xilinx PROM. The additional configuration elements consist of DIL-switches for configuration mode selection, pin headers for external configuration, a push button to initiate configuration and various selection headers to connect the correct programming lines for the chosen mode.

Simple Input/Output is possible with on-board push buttons and LEDs, which are connected to general purpose pins of the FPGA. A LED will be turned on if the according FPGA pin goes high. Using a push button will pull the according FPGA pin high.

More complex communication with the FPGA can be achieved over the parallel port connector. The board features two oscillators. The 4 MHz oscillator output is primarily used for supplying a configuration clock (CCLK) signal for the FPGA and/or PROM. The 20 MHz oscillator output can be used to drive the FPGA's clock nets.

An extension grid is available for soldering additional components to the board. The main grid consists of unconnected pads which allow soldering of through-hole components. Also available are pads with 5 V, 3.3 V or ground potential. Adjacent to the main grid is a row of eleven I/O pads (IO10 - IO20), which are connected to general purpose I/O pins of the FPGA and could easily be attached to additional components.

The size of the bypass capacitors for VCCO should depend on the load of the I/O cell's outputs. For the FPGA-board, conservative values for the average load of all I/O pins have been used ($10\,pF$ per pin). This yielded capacitance values of $4.7\,nF$ per VCCO pin for the high-frequency bypass capacitors. This sizing should limit the drop of the I/O supply voltage to approximately 1-2 %.

When choosing the bypass capacitors for the core supply, the energy requirements of the FPGA have to be considered. For FPGAs which will be used with a fixed design, Xilinx offers the Power Estimator tools. These tools calculate the approximate energy dissipation from the used resources and the operating frequencies of the specific design. For the FPGA-board the high-frequency bypass capacitors have been sized conservatively with a capacitance of $100\,nF$ to ensure a stable power supply for designs with a high power consumption.

Selecting the sizes of mid- and low-frequency bypass capacitors is not as critical as that for high frequencies. The values which are recommended in [58] have been used.

With this setup mainly power-analysis attacks have been performed.

3.5 The FPGA Measurement Board of UCL

When using the power as well as the electromagnetic leakage, the main issue, as underlined in Section 3.2, is to deal with the noise (which could be an important part of the measured signal). For this reason, the UCL board (see Figure 3.3) was built in order to minimize the potential noise sources. It relies mainly on three design principles:

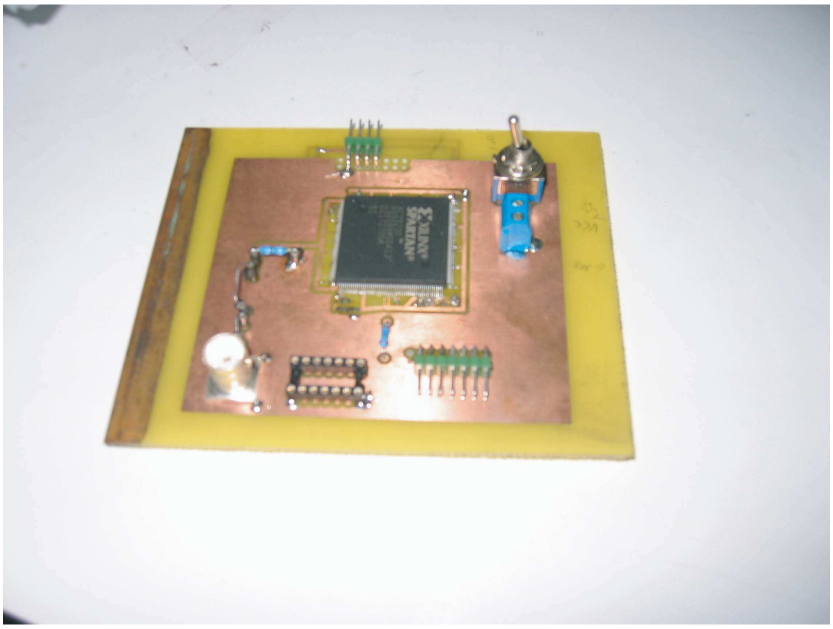


Figure 3.3: UCL measurement setup.

1. Any component (external memory blocks, voltage regulator, ... and passive elements such as capacitors) that are unnecessary for the leakage security evaluation of our targeted designs (mostly block ciphers) were removed. This was motivated by the following (previously mentioned) facts: (1) external components may yield unwanted signals (*e.g.* a voltage regulator usually contains an oscillator) and (2) passive components (*e.g* capacitors) filter the “true” signal.

2. Usual probes (*e.g.* 1 MΩ) can produce disturbances on the signal because of some reflection problems but also because of their structure. For example, a simple voltage probe is usually composed of a main connector with a ground lead attached on it. This small wire loop (around 200 nH) could be stimulated by an outer signal but also by the electromagnetic radiation created by the chip itself. A simple way to reduce this loop is to use the 1-Ohm method [16] proposed by the EMC community. This method is mainly based on the use of BNC connector and two resistors to create a matching impedance network.

3. Another noise source is the power supply itself. Usually the power supply sources are connected to the electrical network which can be highly noisy. Thus, a separated battery was used wired to the board by a coax cable.

The FPGA chosen is the Xilinx SPARTAN II XC2S200-5I with PQ208 package. It has the advantage to be very cheap (less than 30 euros), to be large enough to support all our targeted crypto primitives and to be soldered by hand on a Printed Circuit Board (PCB).

All the VCCINT pins were connected together directly to the power supply, while the GND pins were connected together to a small resistor (usually chosen small enough to not disrupt the IC supply more than 5%). For the reasons before-mentioned we did not place the small capacitors normally required between the GND pins and the VCCINT pins. The consequence is that our design could not work properly at frequency beyond 15 MHz. The clock was also provided by an external signal generator.

3.6 The FPGA Measurement Board of RUB

The setup shown here is in use for EM measurements at a Xilinx Spartan-3 XC3S200-4FT256C FPGA that is clocked at 50 MHz. In this approach we did not carried out modifications at the FPGA board.

The antenna (no. 2 in Fig. 3.4) is a commercial H-field probe RF B 0,3-3[1]. It is positioned directly on top of the FPGA chip (no. 1). The EM signal measured is processed by a 30 dB antenna amplifier PA 303 at the digital oscilloscope. The communication of the PC used for the control of the measurement set-up with the FPGA is done by using the RS232 interface (no. 4). Further, we programmed an internal trigger signal that is available at an external pin (no. 3) of the FPGA board and used it to trigger the EM measurements.

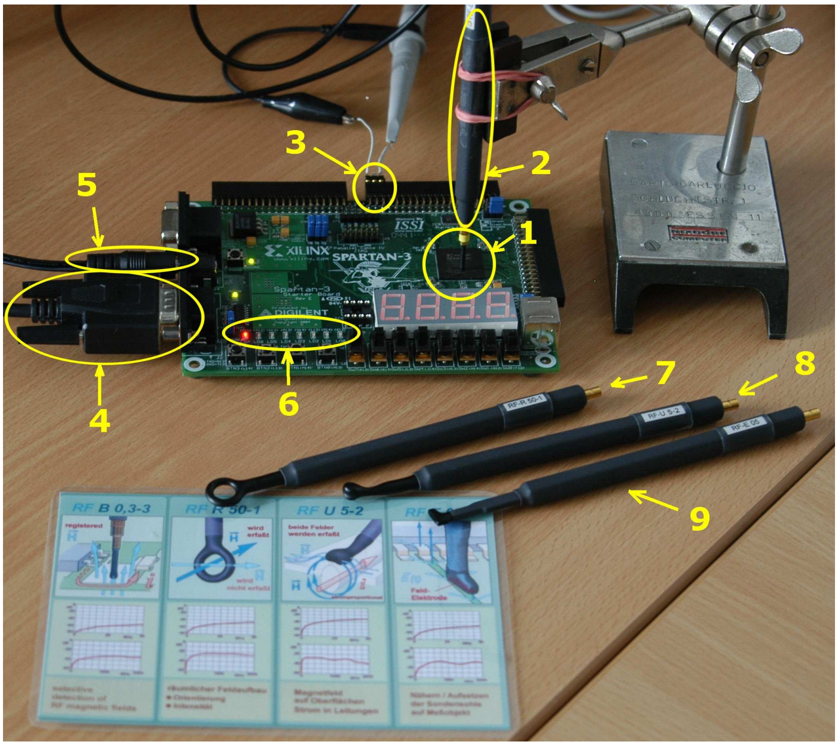


Figure 3.4: RUB measurement setup.

Chapter 4

Power Analysis Attacks against Block Cipher Implementations

4.1 Introduction and Motivation

In this chapter, we intend to collect the main contributions related to the possible side-channel attacks on an FPGA in the framework of the implementation of block ciphers. In practice, we will mainly focus on power analysis attacks since they have attracted the most attention. The chapter is structured as follows.

Section 4.2 investigates correlation analysis attacks against FPGA implementations of block ciphers. We first provide a brief description of our target block cipher and device. Then, we recall the power leakage model that is assumed for an FPGA and we show how an attack based on the correlation coefficient can be successfully mounted. This section aims at providing the reader with a detailed attack procedure, since we believe that the correlation analysis is exemplary of the present state-of-the-art side-channel attacks. We provide simulated and real measurements of a correlation analysis on a simple block cipher.

The correlation coefficient is quite easy to manipulate and permits one to quickly mount an attack. However, it was not the only statistical tool used for these purposes and Section 4.3 intends to quickly recall how the *Difference of Mean test* and the *Maximum Likelihood test* work with appropriate references. It also refers to other possible attack strategies, e.g. based on the square attack [50].

Although very few papers focus on how to counteract side-channel attacks in the specific context of FPGAs, a number of countermeasures in use in the smart card industry can be straightforwardly extended to FPGAs. Section 4.4 specifically focuses on three of them: pipelining (i.e. noise addition), emulation of dynamic and differential logic styles and masking.

Finally, although the masking of block cipher is known as an attractive countermeasure to deflect side-channel attacks, different recent sources showed that it is practically possible (i.e. in a reasonably low number of measurements) to attack such a circuit. In Section 4.5, we summarize a recent result of a higher-order side-channel attack against a masked block cipher. This higher-order attack is also an interesting illustration of a context where maximum likelihood estimation is particularly powerful. We note that other ways to defeat the masking countermeasures have been proposed (e.g. using the glitching activity of the circuits [24]) but have not yet been applied to FPGAs.

4.2 Correlation Power Analysis

4.2.1 Brief description of the target block cipher and device

This chapter investigates the specific case of FPGA implementations of block ciphers. In particular, the Data Encryption Standard (DES, [31]) and Advanced Encryption Standard Rijndael (AES, [32]) will be studied in Subection 4.2.7. For clarity purposes, our theoretical predictions will also be discussed with a simple Substitution Permutation Network. Finally, the devices targeted in this report are Xilinx Virtex® [2] and Spartan® [1] FPGAs.

4.2.2 Selection of a power consumption model

As already mentioned in Section 3.1, most present FPGAs are build from CMOS gates for which the power consumption (and electromagnetic radiation) can be easily predicted thanks to the Hamming distance power consumption model. In the remaining sections, we consequently use the following simple **hypothesis**: "An estimation of the FPGA power consumption at time t is given by the number of bit transitions in the device registers at this time". This hypothesis was successfully used in, *e.g.* [34], [33], [41], [42]. Nevertheless, it is again important to observe that we used a very simple hypothesis, assuming only that we were able to distinguish the number of bit switches in the FPGA. The development of a general treatment of the side-channel leakages (not only for FPGAs) is an interesting scope for further research. As a typical example, we have not tried to distinguish between $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions, although it would result in different needs for efficient attacks and countermeasures. As will be emphasized later in the report, improved power consumption models and measurement techniques could be considered and consequently increase the actual efficiency of the resulting power analysis attacks.

4.2.3 Prediction of the device power consumption

Based on the previous hypothesis, an attacker may estimate the power consumption of a cryptographic implementation by simply predicting the number of bit transitions in the device registers. This can be done using a selection function D that we define as follows. Let X_i and X_{i+1} be two consecutive values inside a target register (*i.e.* the register values during two consecutive clock cycles). An estimation of the target register power consumption at the time of the transition between these values is given by the function $D = H(X_i \oplus X_{i+1})$, where $H(x)$ is the Hamming weight of a bit vector x. An attacker who has to predict the transitions inside the registers of an implementation therefore needs to answer two basic questions:

1. Which register transitions can we predict?
2. Which register transitions do leak information?

Answering these questions determines which registers will be targeted during the attack. We formalized these questions with two definitions that we illustrate on the simple block cipher of Figure 4.1. Our target encryption network is a reduced version of the Khazad block cipher [5], where the S blocks represent small 4×4 non-linear substitution boxes, the P blocks represent 8-bit permutations (*i.e.* wire crossings), the D layer is a linear diffusion layer and $\oplus$ is a bitwise key addition. In addition, the grey boxes represent the registers inserted in order to pipeline the design. Remark that due to the pipeline structure, one encryption of this block cipher is performed in 9 clock cycles.

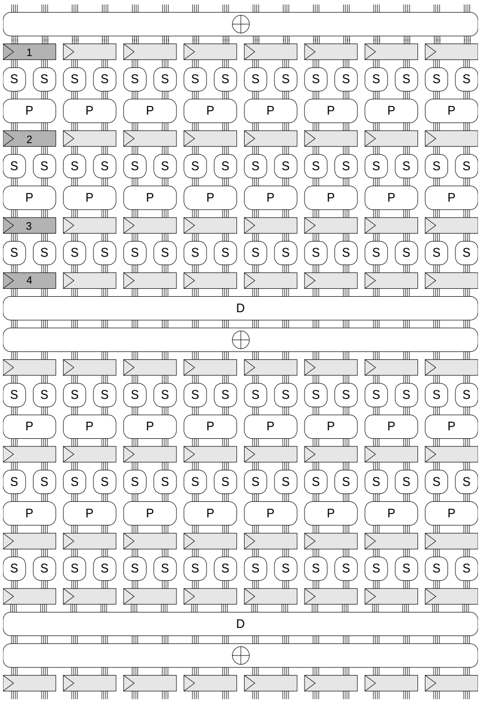


Figure 4.1: Target encryption network.

The definitions are as follows:

- The *predictability* of a register is related to the number of key bits one must know to predict its transitions. For block ciphers, this depends on the size of the S-boxes and the diffusion layer. In practice, it is assumed that it is possible to guess up to 16 or 32 key bits, and the diffusion layer usually prevents the guessing of more than one block cipher round. For example, the dark grey registers in Figure 1 are *predictable* (as all the other registers before the diffusion layer).
- We denote a register as a $full$ (*resp. empty*) register if its transitions leak (*resp.* do not leak) secret information. For example, it is obvious that an input (*resp.* output) register does not leak any secret information as it only contains the plaintext (*resp.* ciphertext). However, a consequence of our prediction model is that the registers following an initial (*resp.* final) key addition do not leak information either. Indeed, the register transitions

after an initial key addition can be expressed as:

$$W_H(input_1 \oplus key \oplus input_2 \oplus key) = W_H(input_1 \oplus input_2)$$

Therefore, the transitions in register 1 (see Figure 4.1) do not depend on the key and this register is *empty* (as all the registers before the first layer of S-boxes). We note that this observation strongly depends on the power consumption model in use and is not true in general.

Based on these definitions, the prediction of a device power consumption takes place as follows.

Let N be the number of plaintext/ciphertext pairs for which the power consumption measurements are accessible. Let K be the secret encryption key. During the prediction phase, the attacker selects the target registers and clock cycle for the previously defined selection function D. Then, he predicts the value of D (*i.e.* the number of bit switches inside the target registers in the targeted clock cycle) for the g possible key guesses and N different plaintexts. The result of this prediction phase is an $N \times g$ **selected prediction matrix**.

In our example, the grey registers 2, 3 and 4 are *predictable* and *full*. As these registers are 8-bit long, the matrix contains numbers between 0 and $3 \times 8 = 24$ and the number of key guesses necessary to predict these transitions is $g = 2^8 = 256$. Remark that we selected these registers for illustration purposes and any set of *predictable* and *full* registers can be used to mount an attack. In addition, targeting registers 2,3 and 4 only allows to obtain eight key bits and a complete key recovery involves to repeat the predictions for the other key bits. In Figure 4.1, there are eight parallel S-boxes and therefore eight prediction steps will be necessary.

For theoretical purposes, it is finally interesting to define the $N \times 1$ **global prediction vector** that contains the number of bit switches inside all the device registers, in the targeted clock cycle for N different plaintexts. This is only feasible if the key is known (*i.e.* when simulating the attacks). In our example, the design contains $8 \times 9 = 72$ 8-bit registers, and the global prediction vector values are between 0 and $8 \times 72 = 576$.

4.2.4 Measurement of the device power consumption

During the measurement phase, the attacker lets the device encrypt the same N plaintexts with the same key, as it was done during the prediction phase. While the chip is operating, he measures the power consumption for the different encryptions and stores the power consumption value for the targeted clock cycle[1]. As a result of the measurement phase, the attacker obtains an $N \times 1$ **global consumption vector** with the values of the power consumption during the targeted clock cycle, for N different plaintexts.

[1] Measurement setups for power analysis attacks have already been intensively described in the open literature. A usual method is to observe the voltage variations over a small resistor inserted in the supply circuit of the cryptographic device. Some averaging is often used to reduce the noise in measurements. However, improved methods exist and consequently improve the attack efficiency. Using more accurate models for the measurement setup is another scope for further research.

4.2.5 Correlation analysis

In the final phase of a power analysis attack, the attacker compares the theoretical predictions of the power consumption with its real measurements. For this purpose, a practical solution, used in several papers and intensively discussed in [9], is to compute the correlation coefficient between the global consumption vector and all the columns of the selected prediction matrix (corresponding to all the g possible key guesses). If the attack is successful, it is expected that only the correct key guess leads to a correct prediction of the power leakage and thus to a high correlation value.

An efficient way to perform the correlation between theoretical predictions and real measurements is to use the Pearson coefficient (see [15]). Let $M(i)$ denote the ith measurement data (*i.e.* the *ith* trace) and M the set of traces. Let $P(i)$ denote the prediction of the model for the ith trace and P the set of such predictions. Then we calculate:

$$C(M,P) = \frac{\mu_{M.P} - \mu_M.\mu_P}{\sigma_M.\sigma_P} \tag{4.1}$$

where μ_M denotes the mean of the set of traces M and σ_M^2 its variance. If this correlation is high, it is usually assumed that the prediction of the model, and thus the key hypothesis, is correct.

4.2.6 An illustrative attack

This subsection illustrates our descriptions with some experiments performed against an FPGA implementation of the block cipher represented in Figure 4.1.

An attack using simulated data

In the attack using simulated data, we chose $N = 1000$ random plaintexts and one secret key and we produced the selected prediction matrix and global prediction vector, as defined in the previous section. Thereafter, we performed the correlation phase between these two matrices. As the relevant information to determine is the minimum number of plaintexts necessary to extract the correct key, we calculated the correlation coefficient for different values of N: $1 \leq N \leq 1000$. In order to underline the importance of clearly setting the attacker capabilities, we also considered two experiments. A first one where the selected prediction matrix contained the transitions in register 4 only (in Figure 4.2) and a second one where it contained the transitions in registers 2, 3 and 4 (in Figure 4.3). We can observe in the figures that both attacks are successful, but the second experiment is significantly faster. In practice, the required number of plaintexts is about respectively 600 and 300, confirming that different attacker capabilities (*i.e.* different knowledge of the design details) may yield different threats.

An attack using measured data

When attacking a device practically, the selected prediction matrix stays unchanged (we predicted transitions in registers 2, 3 and 4, as in Figure 4.3) while we replace the global

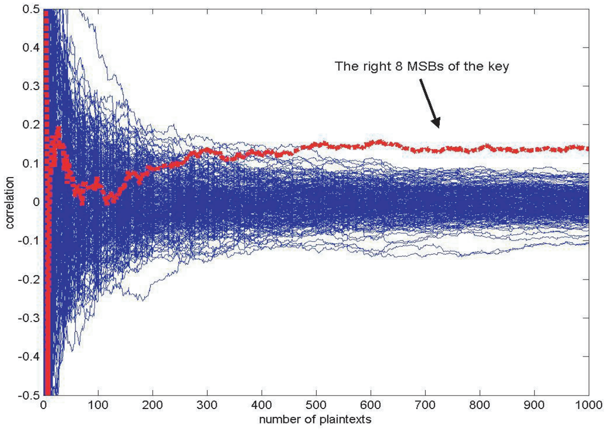


Figure 4.2: A simulated attack using predictions for register 4 only.

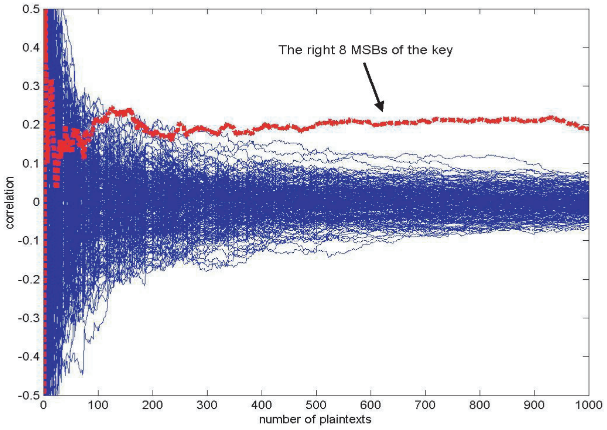


Figure 4.3: A simulated attack using predictions for register 2,3,4.

prediction vector by the global consumption vector. Therefore, we let the FPGA encrypt 2000 plaintexts with the same key as we did in the previous section and produced the matrix as described in Subsection 4.2.3.

To evaluate the quality of our measurements, we made a preliminary experiment and computed the correlation coefficient between the global prediction vector and the global consumption vector, for different number of measurements: $1 \leq N \leq 2000$. As illustrated in Figure 4, the correlation between both vectors is approximately 0.45, confirming our hypothesis to provide a reasonable estimation of the device power consumption. Also, the correlation is not perfect (*i.e.* equal to one), confirming that the power consumption model is not perfect. As already suggested in Section 4.2.2, improved models and measurement tools could be considered, *e.g.* using simple signal processing techniques to improve the quality of the results. As an illustration, in [40], the use of averaging and filtering is investigated and some more specific power consumption models are proposed.

In order to identify the correct key guess, we used the correlation coefficient again. As

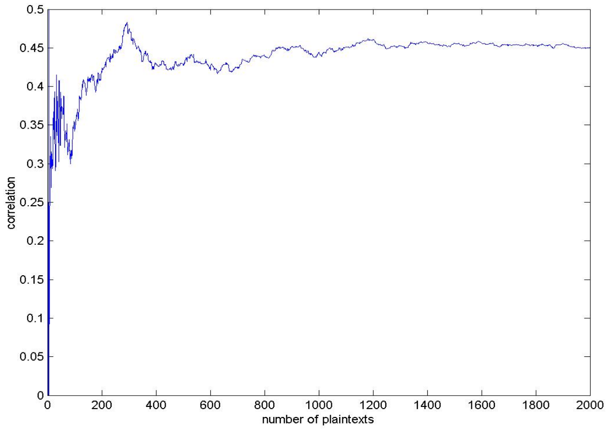


Figure 4.4: Preliminary experiment.

it is shown in Figure 5, the correct key guess is distinguishable after about 1200 traces. As a consequence, the attack is practically successful, *i.e.* the selected prediction matrix is sufficiently correlated with the real measurements and we can extract key information.

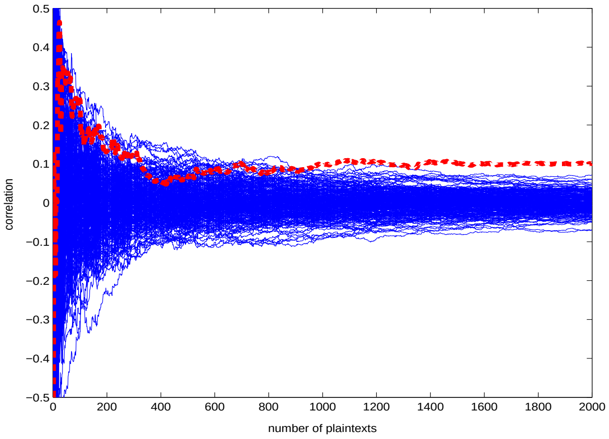


Figure 4.5: An attack using real measurements.

4.2.7 Attacks targeting standard algorithms

The techniques described in the previous sections have been successfully applied to a variety of cryptographic algorithms, including the DES in [41] and AES Rijndael in [33], [42]. In particular, reference [42] relates the security of an implementation to efficiency considerations and evaluates the effect of pipelining and unrolling techniques in this context. It is notably demonstrated that pipelining a loop implementation does not provide an effective countermeasure if an attacker has access to the design details because most of the registers in the pipeline remain predictable. On the other hand, the combination of pipelining and unrolling techniques may counteract power analysis attacks as a random noise generator, because only the outer rounds of such an implementation can then be predicted.

A particular advantage of the correlation power analysis used in these references is the possibility to obtain "theoretical predictions" of the attacks, using simulated data. However, in practice, these predictions require the computation of a fastidious amount of correlation values (typically $g \times N$, where g is the number of key guesses considered) and are specific to one single implementation, device, secret key and selection of plaintexts. As a consequence, a statistical approach to evaluate a circuit security would be relevant.

A theoretical evaluation of power analysis attacks and countermeasures in the smart card context was proposed in [23]. A similar approach applied to FPGAs can be found in [44]. Both papers allow to evaluate the required number of measurements to break an implementation in function of a number of statistical parameters.

4.3 Other Statistical Tools and Attacks

The correlation coefficient proposed in the previous section and used to successfully attack FPGA implementations is not the only possibility when choosing a method to recover a secret data from the information emanated from a secure item (presently, an FPGA). Among the signal processing literature there exist many statistical tools that may help to reveal the secret with more or less efficiency, depending on the attack context.

For example, in the original paper describing a power analysis [21] the authors used a *Difference of Mean test* in their attack procedure. More methods are cited in [13]. The *Difference of Mean test* is easy to implement and was first adopted because it allows an attacker to carry out an attack without any knowledge of the target cryptosystem implementation. For more detailed description of an attack using this tool we refer the reader to the few following papers [21, 12, 6].

However neither the correlation analysis nor the difference of mean test are optimal tools to exploit side-channel leakages. For that reason, [3] and more recently [37] investigated a *Maximum Likelihood* approach. Under the assumption that the noise linked to the signal has a Gaussian distribution, the *Maximum Likelihood hypothesis test* can be efficiently applied to side-channel attacks and yields a better efficiency: according to [3], using the latter allows the authors to successfully guess the secret key with two times fewer measurements compared with a distance of mean approach. Again more details can be found in the papers aforementioned or in a signal processing book [19].

In addition to the choice of the statistical tool, it must be observed that different strategies can be adopted for the key hypothesis step in a side-channel attack. The most basic strategy, as previously presented, makes a key hypothesis on the input of one substitution box and tries to predict the power consumption at the output of this S-box. Alternative solutions may exist, for example by taking advantage of existing crytanalytic techniques. A noticeable example is the work of Carlier et al. [50] where a side-channel extension of the square attack is proposed. Basically, square attacks are based on the propagation of "active" (roughly meaning "taking every possible values") and "passive" (i.e. fixed) bytes through a cipher. As a matter of fact, passive bytes consume no power which yields an immediate side-channel distinguisher.

4.4 Countermeasures

4.4.1 Pipelining

A straightforward idea to make the power analysis more difficult is to bury the useful signal into added noise. [41] explained that unrolling and pipelining the implementation of block ciphers may help to hide the signal into the noise produced by the unpredictable parts of the circuits. Let us recall that the general framework of a power analysis is to predict the electrical behavior of the device with a small part of the whole secret key. The decision in favor of a particular key guess is made depending on whether the predicted behavior corresponds to the observed one. FPGAs are (at least for the recent ones) sufficiently big enough to support a complete unrolled and pipelined block cipher design. As a consequence, an attacker can only predict a small part of the electrical behavior of the chip. This fact was aforementioned in the Subsection 4.2.3 where the notion of *predictability* of a register was introduced.

Consequently, when only one register is predicted, we need significantly more traces than when several registers are predicted. The efficiency of an attack against a loop implementation is notably due to the fact that most registers are predictable, because only one round is implemented. In case of unrolled and pipelined implementations, the situation strongly differs, as only the outer rounds are partially predictable. As a consequence, the inner rounds may be viewed as noise generators and therefore act as a well known DPA countermeasure. Although noise addition does not fundamentally counteract power analysis attacks (the signal is still present and may still be recovered), it has the advantage of decreasing the correlation between predictions and measurements. Moreover, if the noise is added in the form of unrolled pipeline stages, it does not reduce the efficiency of an implementation. Finally, the correlation power analysis recalled in the Section 4.2, allows us to show the effect of unrolled and pipelined architectures on resistance against DPA. The results are displayed in Figure 4.6.

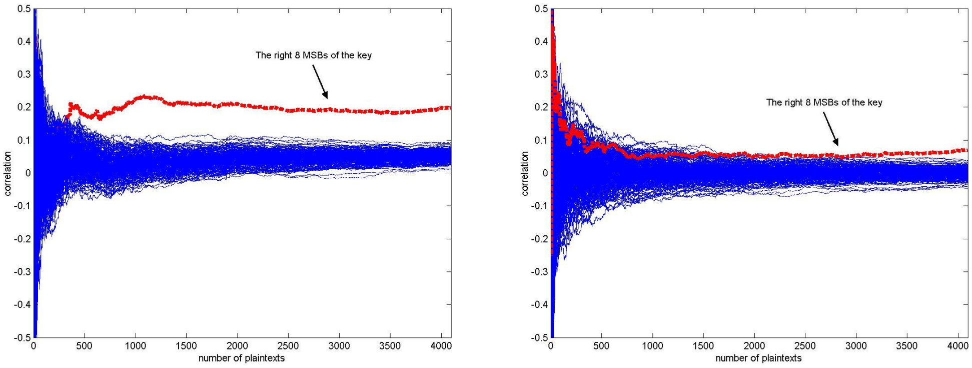


Figure 4.6: Attacks against loop and unrolled FPGA implementations of the AES Rijndael.

4.4.2 Emulating dynamic differential logic in FPGAs

[45] and [46] introduced the idea of using dynamic and differential logic (DDL) styles as a way to counteract power analysis attacks. Implementing an encryption module with such a logic style permits a designer to increase security against power analysis since the resulting circuits consume an amount of power that is supposed to be independent of the data handled[2]. Examples of logic styles proposed and evaluated for these purposes are the Sense Amplifier Based Logic (SABL) [45] or Dynamic Current Mode Logic (DyCML) [22]. However, a number of alternative solutions exist in the circuit literature and were previously proposed for cost, speed or efficiency reasons.

In 2004, [47] proposed to extend such a principle to an FPGA design methodology. The adopted strategy was to design new standard cells by combining building blocks from an existing standard cell library or from a slice to make new compound standard cells, which mimic the behavior of the SABL gates. Improved results are presented in [48].

In practice, most of the attacks that were carried out on FPGA were usually targeting the output of one (some) sensitive register(s). The reason is that determining the moment on your sampled consumption trace that corresponds to a particular consumption is not an easy task. As registers are clocked, the moment the data is loaded is easy to determine and very systematic. As a consequence, the simplest way to balance the consumption is to add to each sensitive register another register that has an inverse behavior. In this way, when a sensitive register consumes (resp. does not consume) an amount of charge, its twin register does not consume (resp. consumes) the same quantity of charge.

It is clear that such a naive solution does not yield a perfect security, since an adversary can still target the logic between any two registers. However, this example allows us to illustrate that the security against side-channel attacks is usually a tradeoff between simplicity, efficiency and security. Typically, the duplication of balanced registers can be easy to implement in an FPGA design and may improve security. A more elaborated solution like the one in [47] improves security further at the cost of a more complex design process and of an expensive final implementation. Even better solutions require to balance not only the logic elements but also the routing in a design. At no point we have a perfect (i.e. theoretical) security but any of these solutions increases the difficulty of performing the attacks in practice. In general, the evaluation of the proposed countermeasures against side-channel attacks (not only DDLs) is still a matter of further research.

4.4.3 Masking

The idea of masking the intermediate values inside a cryptographic algorithm is suggested in several papers as a possible countermeasure to power analysis [4, 11, 14, 36]. The technique is generally applicable if all the fundamental operations used in a given algorithm can be rewritten in a masked domain. This is easily seen to be the case in classical algorithms such as the DES [31] or AES [32]. In the next section, we aim to discuss the security of masked

[2] In practice, some parasitic capacitances cause the existence of a data-dependent behavior even for these logic styles. However, it is commonly admitted that the leakage variations of these circuits are significantly smaller and consequently more difficult to exploit.

FPGA implementations. For this purpose, we start by giving a simple description of our target designs. An unmasked block cipher design is represented in Figure 4.7, where the b_is represent known input values, the k_is are the secret encryption key bits and the S blocks are non-linear substitution boxes (let N_s be the number of such S-boxes). In accordance with the structure of most present block ciphers [8, 5, 31, 32], we do not loose in generality by focusing our attention to this combination of key additions and non-linear S-boxes. Remark that the bit-widths are not specified on the scheme.

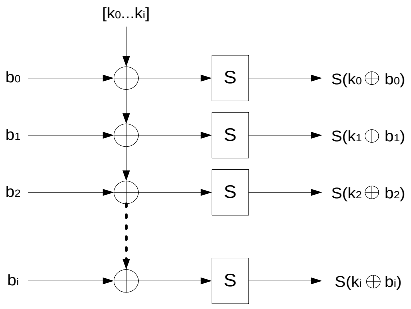


Figure 4.7: Unprotected scheme.

Our protected implementation is represented in Figure 4.8. The masking principle is as follows. After having XORed the random mask to the initial data, both the mask and the masked data are sent through a non-linear S-box. S is the original S-box from the algorithm and S' is a precomputed table such that we have:

$$S(b \oplus k \oplus r) = S(b \oplus k) \oplus S'(r, b \oplus k \oplus r) = S(b \oplus k) \oplus q$$

As a consequence, the output values are still masked with a random mask q. Note that we considered a masking scheme at the substitution box level although a mask can be applied at any possible level. In general, masking small operations is preferred for efficiency purposes because the table required to track the masked data (e.g. S' in our example) is smaller. For example, masking at the gate level has been suggested in [49].

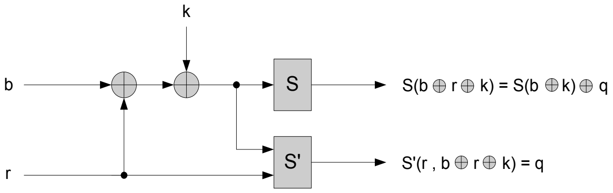


Figure 4.8: Masked scheme.

4.5 Higher-Order Attacks

Higher-order side-channel attacks against masked implementations have been introduced in [27] and are a general way to target such protected designs. Although the original attack seemed somewhat specific and applicable only to certain smart card implementations, [51] suggested that higher-order power analysis is possible, without any additional hypothesis than usually assumed for first-order attacks. They proposed a way to combine the leakages corresponding to the masked data and its mask even if their respective position within the sampled data is unknown. Subsequently, [43] proposed an extension of these attacks by considering a more general power consumption model (corresponding to a FPGA consumption model). But although these papers provide indications for the practical implementation of the attack, the number of observations required to retrieve the secret key is generally large (at least significantly larger than in a first-order power analysis attack). In this section, we will describe a more recent results of higher-order attacks with FPGA experiments [37]. The proposed technique is based on the efficient use of the statistical distributions of the power consumption in an actual design and yields better efficiency than previously proposed attacks.

Remark that we focused our attention on this recent result since it is the only higher-order attack that has been specifically applied to FPGAs. However, as already mentioned, there exist other solutions to defeat the masking countermeasure, e.g. based on the glitching activity of a circuit [24]. The application of these techniques to FPGAs is a scope for further research.

4.5.1 Attack Procedure and Details

Let us describe the proposed technique of [37] with the single S-box scheme of Figure 4.9, where the inputs b, r and k are N_b-bit wide. First, we express the power consumption of one pair of S and S' boxes in case of a pipeline block cipher implementation and denote it as a random variable O, standing for observations. That is, we assume that the structure displayed in Figure 4.9 is fed with a new input at each clock cycle. As explained in the previous section, the power consumption is a function of any two consecutive values. If $b \oplus k$ switches into $b' \oplus k$ and q switches into q', we have:

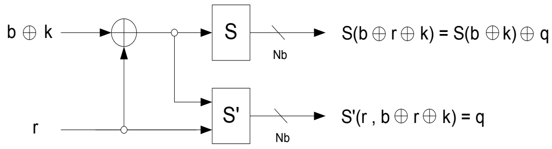


Figure 4.9: A masked 4-bit S-box

$$O = WH\Big[\big(S(b \oplus k) \oplus q\big) \oplus \big(S(b' \oplus k) \oplus q'\big)\Big] + WH\Big[q \oplus q'\Big]$$

Defining the random variable $\Sigma = S(b \oplus k) \oplus S(b' \oplus k)$, where Σ stands for secret state and the random variable $R = q \oplus q'$, where R stands for random state, it is therefore possible to rewrite the observations as:

$$O(\Sigma, R) = WH\big[\Sigma \oplus R\big] + WH\big[R\big]$$

Remark that the operator used to combine the two leakage contributions is a '+' because in our analysis, the masked data and its mask are loaded on the register at the same time. But in other contexts, we may choose a '$-$' as in [27], or a '$\times$' as in [51, 43]. Actually, no matter what operator we use, the main point is to gather the two (or more in case of higher-order masking) statistical distributions of the power consumption so that the combined statistical distribution is key-dependent.

Indeed, while it is not possible to predict the observations, because they depend on unknown mask and key values, we can still analyze their statistical behavior. For a fixed value of the secret state $\Sigma = \sigma_i$, we can determine all the possible observations, for all the different possible random states $R = r_j$. From this analysis, it is therefore possible to derive the probability density functions $P[O = o_i|\Sigma = \sigma_i]$, for all the possible secret states.

In practice, because the observations are a sum of two Hamming weight values, they are distributed as binomials and the number of possible distributions for $P[O|\Sigma = \sigma_i]$ equals $N_b + 1$. As a simple illustration, if $N_b = 4$, the five possible distributions of the observations are given in Figure 4.10.

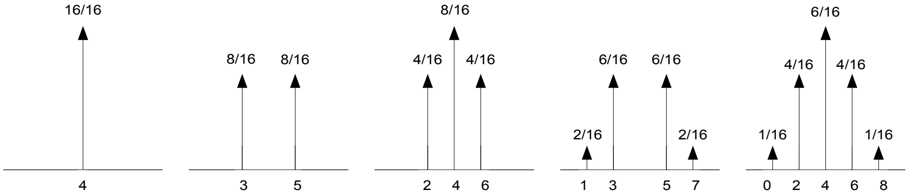


Figure 4.10: Probability density functions $P[O|\Sigma = \sigma_i]$ with $N_b = 4$.

The important consequence is that, knowing a secret state σ_i, we know the probability of making an observation o_i. This provides us with the tool to mount a new attack, based on a maximum likelihood approach.

Remark: The distributions $P[O|\Sigma = \sigma_i]$ all have the same mean value, $E(O|\Sigma = \sigma_i) = N_b$ and only differ in their variances. This fact allows to understand the origin of previous attacks, as the one in [51], where it is proposed to square the power consumption traces in order to obtain key-dependent measurements. The reason is that the mean of the squared power trace is a function of the mean and the variance of the initial power trace:

$$E\Big((O|\Sigma = \sigma_i)^2\Big) = E\Big((O|\Sigma = \sigma_i)\Big)^2 + V(O|\Sigma = \sigma_i)$$

It is also clear that the information contained in the expectation of the squared power trace is poor compared to what can be obtained using the complete statistical distribution of the observations.

Now, using the usual framework of side-channel attacks, we would like to find the secret key k, using a serial of observations $o_1, o_2, ..., o_n$, obtained by feeding the encryption device with a serial of input texts $b_0, b_1, ..., b_n$ (the input transition $b_0 \rightarrow b_1$ gives rise to the observation o_1).

For this purpose, we first remark that, knowing the sequence of input texts $b_0, b_1, ..., b_n$, each key candidate $k_i \in [0, 2^{Nb} - 1]$ specifies one sequence of secret states. Therefore, we have 2^{N_b} possible chains of states denoted as:

$$\begin{aligned}
\Sigma^*(k_0) &:= \{\sigma_1(k_0), \sigma_2(k_0), ..., \sigma_n(k_0)\};\\
\Sigma^*(k_1) &:= \{\sigma_1(k_1), \sigma_2(k_1), ..., \sigma_n(k_1)\};\\
\Sigma^*(k_2) &:= \{\sigma_1(k_2), \sigma_2(k_2), ..., \sigma_n(k_2)\};\\
&\qquad\qquad\qquad\qquad\qquad\qquad ...
\end{aligned}$$

In practice, these state sequences cannot be observed directly, but only through the power consumption of the device, *i.e.* the sequence of observations $O^* := \{o_1, o_2, ..., o_n\}$. Then, for each possible secret state chain, we compute the probabilities $P[O^*|\Sigma^*(k_j)]$. Assuming that the observations are independent (which is reasonable since the attacker feeds the devices with random input texts), it yields:

$$\begin{aligned}
P[O^*|\Sigma^*(k_0)] = &\quad P[O = o_1|\Sigma = \sigma_1(k_0)] \times P[O = o_2|\Sigma = \sigma_2(k_0)] \times ...\\
P[O^*|\Sigma^*(k_1)] = &\quad P[O = o_1|\Sigma = \sigma_1(k_1)] \times P[O = o_2|\Sigma = \sigma_2(k_1)] \times ...\\
P[O^*|\Sigma^*(k_2)] = &\quad P[O = o_1|\Sigma = \sigma_1(k_2)] \times P[O = o_2|\Sigma = \sigma_2(k_2)] \times ...\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad ...
\end{aligned}$$

The chain with the highest probability gives us the most likely key. That is, if the attack is successful, the correct key corresponds to:

$$\underset{\forall\ k_j}{argmax}\, P[O^*|\Sigma^*(k_j)]$$

We note that the proposed approach is similar to the one in [18], where it is demonstrated that Hidden Markov Models may be of great help to describe discrete time processes where a state sequence is hidden. Remark finally that, in order to keep the probabilities $P[O^*|\Sigma^*(k_j)]$ within practical boundaries (for large n's, these probabilities are smaller than the machine-ϵ), we use a step by step normalization (for more details see [37]).

4.5.2 Experimental Results

A practical attack against an FPGA implementation of the scheme in Figure 4.8 was realized, with $N_s = 8$ S-boxes that work in parallel [3]. Our target device was a Xilinx Spartan II FPGA [1] and the random mask values r_i's were generated with an on-chip LFSR. As illustrated in Figure 4.11, the attack is successful after roughly 12 000 measurements. We refer again to [37] for the comparisons of this result with other higher-order attacks.

4.6 Electromagnetic Attacks

To the best of our knowledge, there has only been one public contribution to apply an EMA to FPGA implementations of block ciphers in the previously mentioned square EMA [50].

[3] Due to area constraints, we did not target a standard algorithm such as the AES Rijndael. Indeed, as already mentioned, *e.g.* in [35, 36], the hardware cost of masking a block cipher is a real concern for efficient hardware implementations.

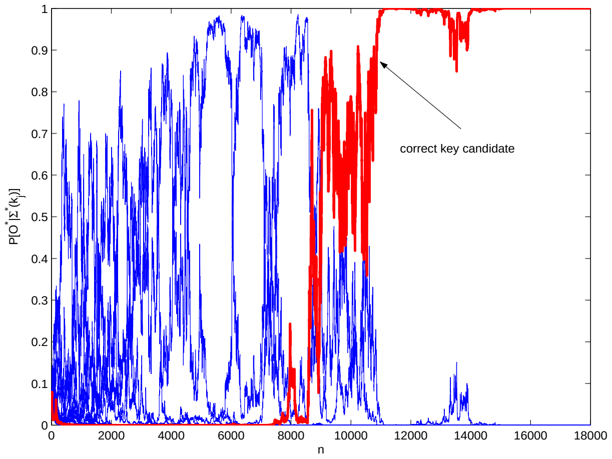


Figure 4.11: A real attack against a masked FPGA design with $N_s = 8$.

However, EMA has been applied to elliptic curve implementations (see the next chapter) and similar measurements could be applied to block cipher implementations as well. Note that the performed electromagnetic measurements only monitor the radiation of a whole device, without trying to take advantage of localization effects, as it has been done in the smart card context. As a matter of fact, localized electromagnetic measurements would require to depackage an FPGA and has not yet been investigated (again, to the best of our knowledge).

Chapter 5

Electromagnetic Analysis of Elliptic Curve Cryptosystems

In this chapter results will be shown of simple electromagnetic analysis (SEMA) on elliptic curve cryptography (ECC) in Sect. 5.2 and of differential electromagnetic analysis (DEMA) in Sect. 5.3.

5.1 Electromagnetic radiation

The current consumption of CMOS circuits is data-dependent. However, for the attacker, the relevant question is to know whether this data-dependent behavior is observable.

The current that flows during the switching of the CMOS gates, causes a variation of the electromagnetic field surrounding the chip that can be monitored by for example inductive probes which are particularly sensitive to the related impulse. The electromotive force across the sensor (Lentz' law) relates to the variation of magnetic flux as follows:

$$V = -\frac{\mathrm{d}\phi}{\mathrm{d}t} \quad \text{and} \quad \phi = \iint \vec{B} \cdot d\vec{A},$$

where V is the probe's output voltage, ϕ the magnetic flux sensed by probe, t is the time, $\vec{B}$ is the magnetic field and $\vec{A}$ is the area that it penetrates.

Maxwell's equation based on Ampère's law relates the magnetic field to their origin:

$$\vec{\nabla} \times \vec{B} = \mu\vec{J} + \epsilon\mu\frac{\delta\vec{E}}{\delta t},$$

where $\vec{J}$ is the current density, $\vec{E}$ is the electrical field, ϵ is the dielectric permittivity and μ is the magnetic permeability.

5.2 Simple Electromagnetic Analysis on ECC

This section about SEMA starts with a mathematical background in subsection 5.2.1, continues with a small subsection about simple power analysis (subsection 5.2.2) and ends with the results for SEMA in subsection 5.2.3. The latter is divided in three parts, part 5.2.3 about the ECC point multiplication, part 5.2.3 about the ECC point doubling and part 5.2.3 about the ECC point addition.

5.2.1 Mathematical background

Elliptic Curve Cryptography was proposed independently by Miller [30] and Koblitz [20] in the 80's. Since then a considerable amount of research has been performed on secure and efficient ECC implementations. The benefits of ECC, when compared with classical cryptosystems such as RSA [38], include: higher speed, lower power consumption and smaller certificates, which are especially useful for wireless applications.

An elliptic curve E is expressed in terms of the Weierstrass equation: $y^2 = x^3 + ax + b$, where $a, b \in GF(p)$ with $4a^3 + 27b^2 \neq 0 \pmod{p}$. The point at infinity $\mathcal{O}$ plays a role analogous to that of the number 0 in an ordinary addition. The points on an elliptic curve together with the operation of addition form an Abelian group. Then it is straightforward to introduce the point multiplication as main operation for elliptic curve cryptosystem (ECC).

The attacks described here are performed on the execution of an elliptic curve multiplication. Often, the "double-and-add" method (table 5.1,[26]) is used in which a conditional branch is executed depending on the value of the key-bits. Each iteration one key-bit is investigated, if the key-bit is 0, the intermediate value is doubled by means of an EC point doubling, see the algorithm in table 5.2, and if the key-bit is 1, besides the doubling the intermediate value, the cleartext is added one more time with the EC point addition algorithm, see the algorithm in table 5.2. In the explanation of those algorithms, P is a point of an elliptic curve and is called the plaintext, k is the private key and Q is the ciphertext. As point P, Q is also a point of the elliptic curve.

Table 5.1: EC point multiplication

Algorithm 1: EC point multiplication

INPUT: EC point P, integer $k = (k_{l-1}, k_{l-2}, ..., k_0)_2$
OUTPUT: $Q = [k]P$

```
$Q \leftarrow P$
For i from l-1 to 0 do:
    $Q \leftarrow 2Q$
    If $k_i = 1$ then
        $Q \leftarrow Q + P$
Return $Q$
```

The algorithms for an EC point doubling and EC point addition are described in algorithm 2 in table 5.2 . Both exist out of 14 substeps and each of those fourteen steps executes 1 or 2 operations, a Montgomery multiplication and/or a modular addition. In this design, the Montgomery multiplication takes 500 clock cycles, in comparison, the modular addition only counts 300 clockcycli. Projective coordinates are used, written in these coordinates, a point $P(X, Y)$ becomes $P(X, Y, Z, aZ^4)$.

Table 5.2: EC point addition and EC point doubling

Algorithm 2: EC point addition and EC point doubling

```
INPUT:  $P_1 = (X_1, Y_1, 1, a)$                  INPUT:  $P_1 = (X_1, Y_1, Z_1, aZ_1^4)$
        $P_2 = (X_2, Y_2, Z_2, aZ_2^4)$
OUTPUT: $P_1 + P_2 = P_3$                         OUTPUT: $2P_1 = P_3$
        $= (X_3, Y_3, Z_3, aZ_3^4)$                       $= (X_3, Y_3, Z_3, aZ_3^4)$

1.  $T_1 \leftarrow Z_2^2$                                  1.  $T_1 \leftarrow Y_1^2$            $T_2 \leftarrow 2X_1$
2.  $T_2 \leftarrow X_1T_1$                                 2.  $T_3 \leftarrow T_1^2$            $T_2 \leftarrow 2T_2$
3.  $T_1 \leftarrow T_1Z_2$    $T_3 \leftarrow X_2 - T_2$       3.  $T_1 \leftarrow T_2T_1$           $T_3 \leftarrow 2T_3$
4.  $T_1 \leftarrow Y_1T_1$                                 4.  $T_2 \leftarrow X_1^2$            $T_3 \leftarrow 2T_3$
5.  $T_4 \leftarrow T_1^2$     $T_5 \leftarrow Y_2 - T_1$       5.  $T_4 \leftarrow Y_1Z_1$           $T_3 \leftarrow 2T_3$
6.  $T_2 \leftarrow T_2T_4$                                 6.  $T_5 \leftarrow T_3(aZ_1^4)$      $T_6 \leftarrow 2T_2$
7.  $T_4 \leftarrow T_4T_3$    $T_6 \leftarrow 2T_2$            7.                                $T_2 \leftarrow T_6 + T_2$
8.  $Z_3 \leftarrow Z_2T_3$    $T_6 \leftarrow T_4 + T_6$       8.                                $T_2 \leftarrow T_2 + (aZ_1^4)$
9.  $T_3 \leftarrow T_5^2$                                  9.  $T_6 \leftarrow T_2^2$            $Z_3 \leftarrow 2T_4$
10. $T_1 \leftarrow T_1T_4$    $X_3 \leftarrow T_3 - T_6$       10.                               $T_4 \leftarrow 2T_1$
11. $T_6 \leftarrow Z_3^2$     $T_2 \leftarrow T_2 - X_3$       11.                               $X_3 \leftarrow T_6 - T_4$
12. $T_3 \leftarrow T_5T_2$                                 12.                               $T_1 \leftarrow T_1 - X_3$
13. $T_6 \leftarrow T_6^2$     $Y_3 \leftarrow T_3 - T_1$       13. $T_2 \leftarrow T_2T_1$           $aZ_3^4 \leftarrow 2T_5$
14. $aZ_3^4 \leftarrow aT_6$                                14.                               $Y_3 \leftarrow T_2 - T_3$
15. Return $X_3$                                        15. Return $X_3$
16. Return $Y_3$                                        16. Return $Y_3$
17. Return $Z_3$                                        17. Return $Z_3$
```

5.2.2 For comparison: Simple Power Analysis (SPA)

To point out the difference in measured traces between power consumption and electromagnetic radiation, this section shortly gives the result of a simple power analysis. Fig. 5.1[1] shows the power consumption of an FPGA during execution of the EC algorithm described in section 5.2.1. The envelope of the signal reveals the private key immediately. The cause is the key dependency of the conditional branch in the algorithm. The difference is not only visible in the execution time, but also in used power consumption. An EC point doubling ends with two gaps, for which the explanation will be given in section 5.2.3 and section 5.2.3. An EC point addition does not show these gaps, as a consequence key-bit 0 can be distinguished from key-bit 1 because of their difference in power consumption.

[1] Clock frequency: 300 kHz, Sampling frequency: 25 MS/s, Total duration of measurement: 160 ms

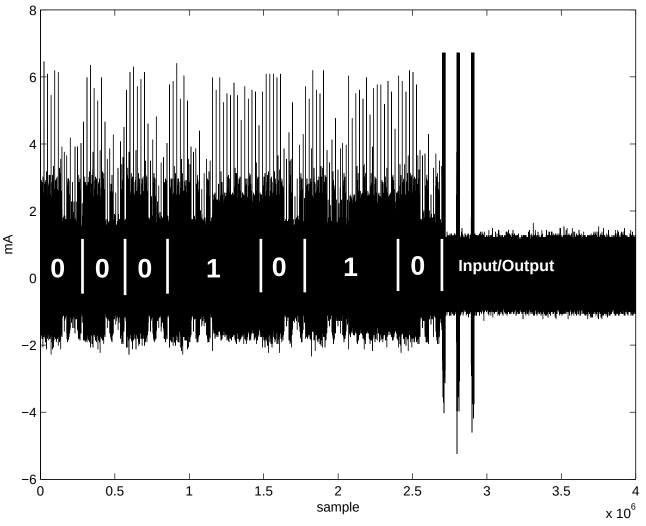


Figure 5.1: The measured power consumption during an EC point multiplication [1]

5.2.3 SEMA Results

EC point multiplication with eight key-bits

A reasoning analogue to a simple power analysis can be applied to the radiation. The different operations result in a different radiation pattern. This way a clear distinction between the execution of an EC point addition and doubling is noticed. By using the knowledge of the algorithm, the key-bits are deducable. Fig. 5.2[1] shows the electromagnetic radiation of an EC point multiplication with eight key-bits during the time of execution. The key-bits are written in the figure.

An EC point doubling

A more profound investigation of the field reveals some other information about the algorithm. The fourteen steps of one operation are visible in Fig. 5.3[2]. If a Montgomery multiplication is performed in a step, a bump is seen in the field. Absence of the Montgomery multiplication results in a gap in the field. In most of the steps, two peaks are visible, those represent the ending of the modular addition and the Montgomery muliplication. The first one needs less clock cycles than the second one as explained in section 5.2.1.

An EC point addition

The electromagnetic radiation of the EC point addition reveals the same information. Fig. 5.4[2] shows the result.

[2] Clock frequency: 300 kHz, Sampling frequency: 100 MS/s, Total duration of the measurement: 11 ms

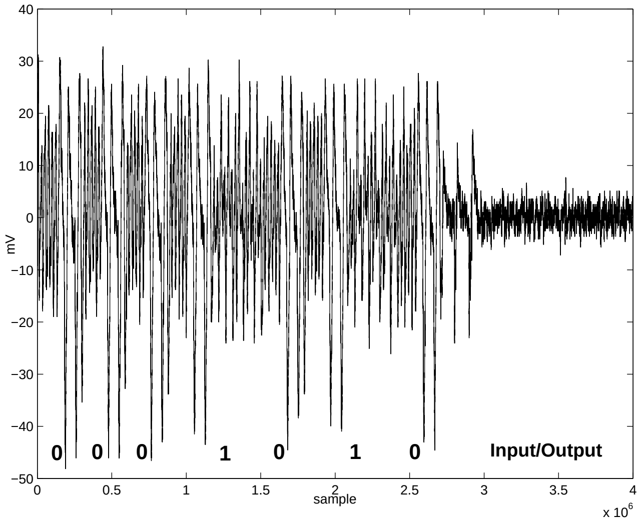


Figure 5.2: The electromagnetic radiation of an EC point multiplication [1]

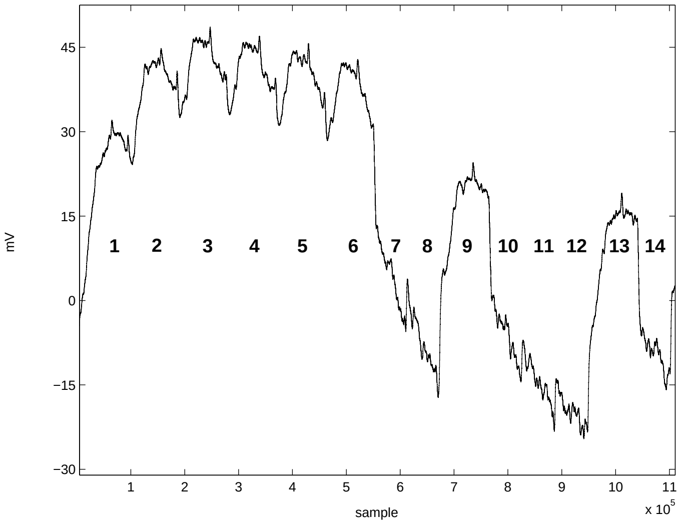


Figure 5.3: The electromagnetic radiation of an EC point doubling [2]

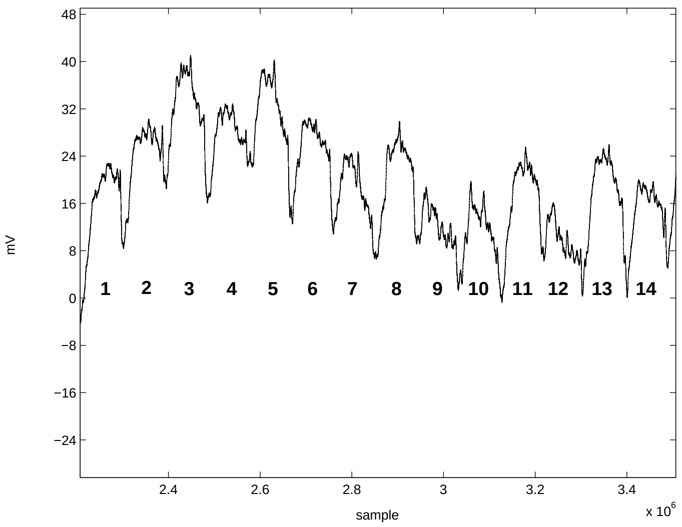


Figure 5.4: The electromagnetic radiation of an EC point addition [2]

5.3 Differential Electromagnetic Analysis

This section starts with the basic mathematical background which is needed to understand the results in subsection 5.3.1, continues with the result of the electromagnetic analysis in subsection 5.3 and ends with the comparable results for a power analysis in subsection 5.3.3 , so both electromagnetic analysis and power analysis can be compared.

5.3.1 Mathematical background

By using the "double-and-add-always" algorithm, a SEMA is countermeasured. The algorithm used is shown in table 5.3. The conditional execution of the EC point addition has disappeared, for a key-bit 1, as for a key-bit 0, an EC point addition is performed after the EC point doubling. The right result, this is depending on the key-bit, is passed through to the next iteration. P is the plaintext, an EC point of the curve, k is the private key and Q the ciphertext.

5.3.2 DEMA Results

This section gives the result of a DEMA. It starts with an explanation of the

Attacking key-bit i

In table 5.4 the sequential behaviour is written down for the second most significant bit (MSB), called 1, and the third MSB, called 2. The difference in bit toggles is depicted in column 3. This difference is the point of attack. The update of the register takes places during exactly one clock cycle. For this reason a good idea of the timing is important.

Table 5.3: EC puntvermenigvuldiging voor DEMA

Algorithm 3: Double-and-add-always algorithm for an EC point multiplication

INPUT: EC point P, integer $k = (k_{l-1}, k_{l-2}, ..., k_0)_2$
OUTPUT: $Q = [k]P$

```
$Q \leftarrow P$
For i from l-1 to 0 do:
    $Q_1 \leftarrow 2Q$
    $Q_2 \leftarrow Q_1 + P$
    If $k_i = 1$ then
        Return $Q_2$
    else
        Return $Q_1$
```

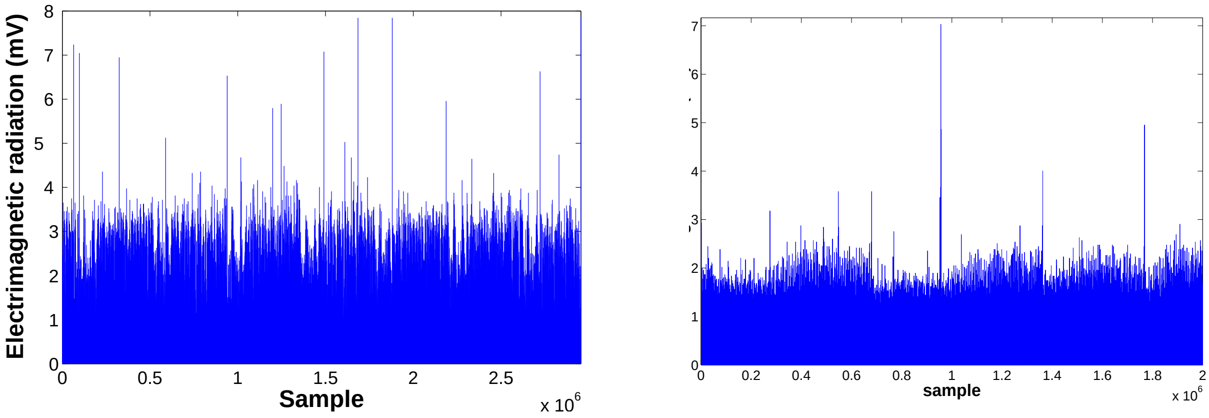
Table 5.4: Attack of key-bit 1

iteration 1	iteration 2	update register Q_1
$Q_1 = 2P$		
$Q_2 = 3P$		
If $k_0 = 1$ **then**		
Return Q_2	$Q_1 = 6P$	$2P \rightarrow 6P$
	$Q_2 = 7P$	
else		
Return Q_1		
	$Q_1 = 4P$	$2P \rightarrow 4P$
	$Q_2 = 5P$	

The measurements

The electromagnetic radiation trace of an EC point multiplication is shown in Fig. 5.5. The highest seven spikes on Fig. 5.5 show the end of seven EC point doubling operations. Our attack point is one of these seven spikes. The first one corresponds to the end of the first EC doubling operation. As shown above this spike shows the end of the second operation which is $Q_1 \leftarrow 2P$ and this step is executed independently from the key bits. The third, fourth and later spikes need the knowledge of the k_{l-2}, k_{l-3} etc. Hence our choice for the measurement point is the second update of Q_1 after the second EC point doubling (Step 3). We use the transitions between the previous value of Q_1, $2P$, and the new value at our target point, $4P$ or $6P$ according to the value of k_{l-2} as the electromagnetic radiation predictions.

As preprocessing technique, we took the maximum of each clock cycle, as can be seen in Fig. 5.6.We have found the maximum value of the measurement data in each clock cycle as $M_2(i,j) = max(M_1(i, D_i \cdot (j-1) + 1 : D_i \cdot j))$, where $i = 1, \ldots, N$, $j = 1, \ldots, 2400$. $M_2(i,j)$



(a) **(b)**

Figure 5.5: Electromagnetic radiation trace of a 160-bit ECPM over $GF(p)$ with Algorithm 3: (a) complete, (b) around the attack point

is the element of the matrix M_2 at the ith row and the jth column. D_i is the number of samples per clock cycle during ith measurement, $M_1(i, D_i \cdot (j-1)+1 : D_i \cdot j)$ is the row vector $[M_1(i, D_i \cdot (j-1)+1) \;\; M_1(i, D_i \cdot (j-1)+2) \;\; \cdots \;\; M_1(i, D_i \cdot j)]$.

We have implemented the EC point multiplication with algorithm 3 in table 5.3 in the C programming language. The C program computes N EC point multiplications with N EC points and the key. The EC points and the key are the same as the ones given to the FPGA during the measurements. During the execution of the EC point multiplications, the C program computes the number of bits that change from 0 to 1 in some registers at the steps corresponding to the five spikes shown in Fig. 5.6. The number of transitions is used as the electromagnetic radiation prediction.

One of the steps of the attack is to find the right steps in the C program to predict the electromagnetic radiation which corresponds to the measurement points. As we want to measure the electromagnetic radiation of the FPGA around the last clock cycle of the second EC doubling in Step 3 of algorithm 3, the fifth peak corresponds to this event. Hence we can predict it by counting the number of transitions between the bits of the coordinates of Q_1 and Q_2; for $k_{l-2} = 0$ we count the number of transitions between $2P$ and $4P$ and for $k_{l-2} = 1$ between $2P$ and $6P$.

We have produced two electromagnetic radiation prediction matrixes, M_3 and M_4, for the $k_{l-2} = 0$ and $k_{l-2} = 1$ guesses, respectively. M_3 and M_4 have one column for the fifth peak and N rows for the N EC points.

Now we can learn the right value of k_{l-2} by finding the correlations between M_3 and M_4 and the column of M_2 which corresponds to the fifth spike in Fig. 5.6. If the correlation between M_3 and M_2 is higher than the correlation between M_4 and M_2, then we decide that $k_{l-2} = 0$, otherwise we decide that $k_{l-2} = 1$. We are also interested in finding the minimum number of measurements that are necessary to find the right key-bit. Figure 5.7 shows the change in correlation between M_3 and M_4 and the column of M_2 which corresponds to the fifth spike in Fig. 5.6 according to the number of measurements used.

It is visible from Fig. 5.7 that the correlation for the prediction for the $k_{l-2} = 1$ guess is

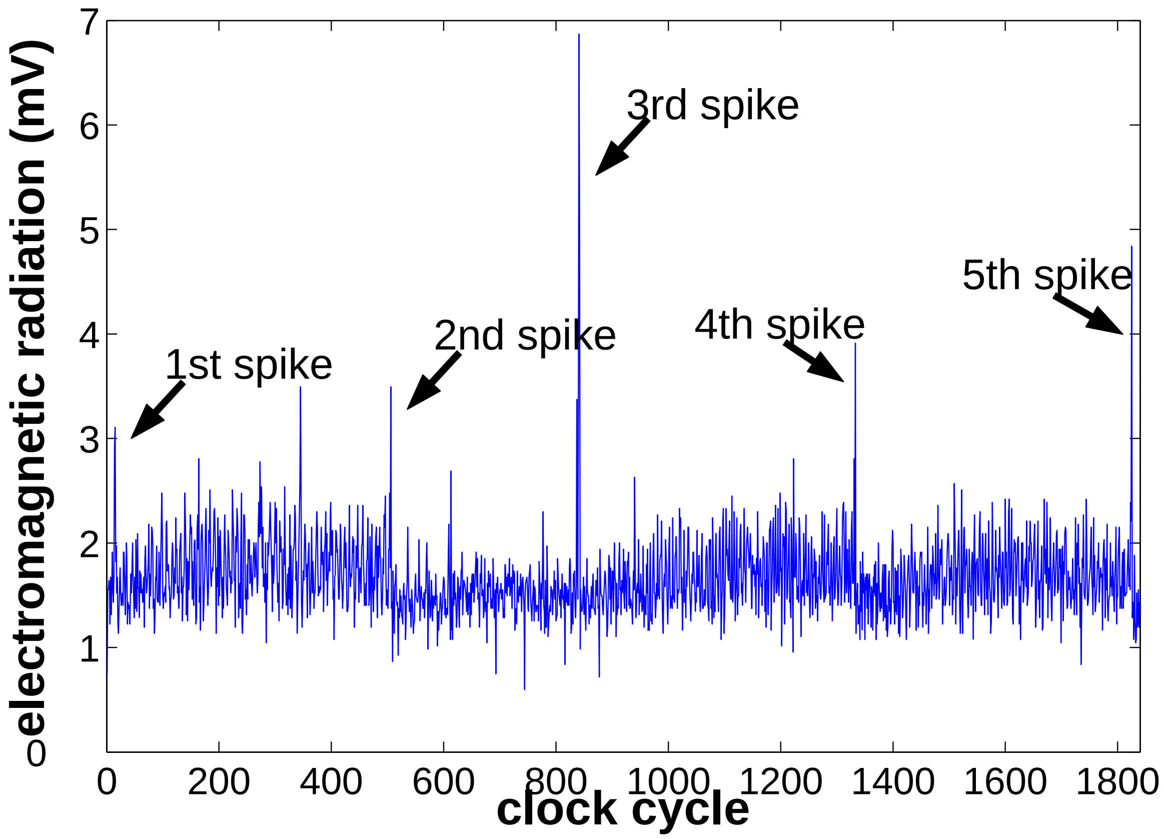


Figure 5.6: Electromagnetic radiation trace of a measurement after taking the maximum value in every clock cycle

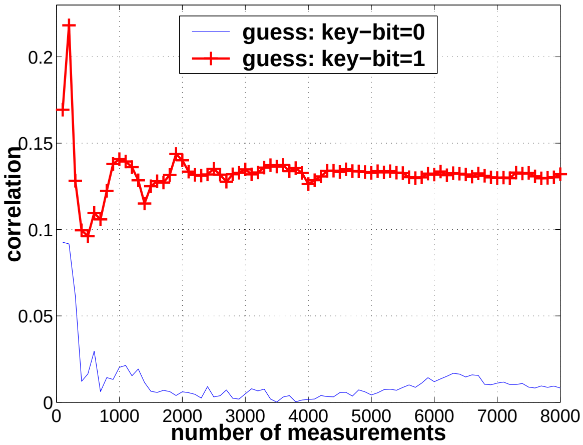


Figure 5.7: Correlation between the electromagnetic radiation measurements and the predictions of the fifth spike in Fig. 5.6 as a function of the number of measurements

higher than the correlation for the prediction for the $k_{l-2} = 0$ guess. By using the first 1000 measurements the decision of $k_{l-2} = 1$ can be made.

5.3.3 For comparison: Differential Power Analysis (DPA)

We did the same preparation for the power measurements and we got the result depicted in Fig. 5.8. As can be seen the correlation for power analysis is higher than for electromagnetic analysis. This is due several reasons. First, the measurement probes used for measuring the electromagnetic field pick up some extra noise coming from the outside world. Next, the selfmade probe used for this experiments is not so accurate as the current probe used for the power measurements. A third reason is the model. As the direction of the current and the location of the current on the chip are parameters which are not used in current electromagnetic field models for differential analysis, the model itself adds some extra noise to the results.

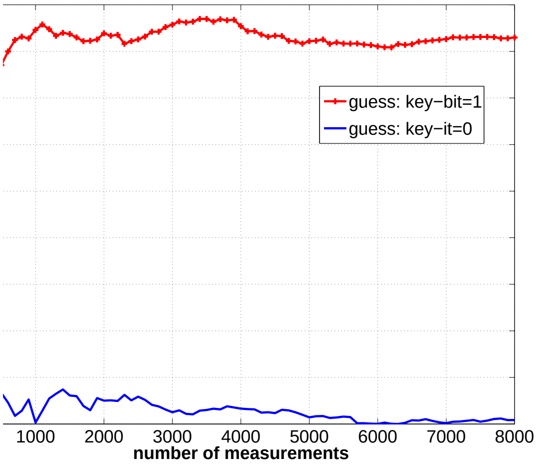


Figure 5.8: Correlation between the current consumption measurements and the predictions as a function of the number of measurements

5.4 Conclusion

The main conclusion from this chapter is that ECC, when no countermeasures are implemented, is easily broken with electromagnetic analysis or power analysis. Electromagnetic analysis, as power analysis is a very powerful tool to attack cryptographic devices. The main advantage of EMA compared with PA is that the attacker can perform the attack from a distance. Another advantage is that it is possible to measure locally, although this is not deployed in the results described. Up to now, the results of DEMA are deterior to those of DPA, because of the reasons summed up in subsection 5.3.3. Some research in these topics could be made to improve the results.

Chapter 6

Conclusion

In this report, we reviewed a number of recent side-channel attacks performed against FPGA implementations of cryptographic algorithms. In particular, block ciphers and elliptic curve cryptosystems were investigated. Although it was initially believed that, due to circuit complexity reasons, such attacks would be difficult to carry out in practice, the presented results underline that FPGAs implementations can be analyzed in a very similar way as smart cards.

In practice, the circuit complexity of FPGAs makes the good prediction of their power consumption more challenging than in the smart card context. However, even with (very) simple models (e.g. Hamming distance based), it is possible to obtain sufficient correlation values between actual measurements and theoretical predictions. Although the obtained correlations are usually lower than those obtained for smart cards, they allow to mount side-channel attacks at the cost of a few more measurements.

The ability to perform parallel computing is another feature that increases the complexity of performing side-channel attacks. However, as already suggested by a number of different sources, such large architectures counteract side-channel attacks in the same way as a noise generator. As a consequence, they only increases the number of measurements required for a successful attack, as measured in this report.

The conclusion is that the security against side-channel attacks has to be considered as a serious threat for FPGAs, as for most CMOS-based microelectronic devices designed for security purposes. The presented results relate to power and electromagnetic analysis, which proved to be both applicable in practice. As a matter of fact, the presented electromagnetic analysis of FPGAs is based on a global measurement of the field, with only low spatial resolution. An interesting scope for further research would be to depackage an FPGA and to monitor the EM with a small probe locally.

Finally, most countermeasures applicable to smart cards can be straightforwardly implemented in FPGAs. A number of solutions to increase the security of FPGAs against side-channel attacks are therefore possible and some of them are discussed in the report. However, as in the context of smart cards, no present solution offers theoretical security and the evaluations of a circuit security is mainly based on experimental evidence. The possibility to build a model to prove the security of any implementation against side-channel attacks is another scope for further investigations.

Bibliography

[1] Spartan 2.5V Field Programmable Gate Arrays Data Sheet. http://www.xilinx.com.

[2] Virtex 2.5V Field Programmable Gate Arrays Data Sheet. http://www.xilinx.com.

[3] Dakshi Agrawal, Josyula R. Rao, and Pankaj Rohatgi. Multi-channel Attacks. In Walter et al. [52], pages 2–16.

[4] Mehdi-Laurent Akkar and Christophe Giraud. An Implementation of DES and AES, Secure against Some Attacks. In Çetin Kaya Koç, David Naccache, and Christof Paar, editors, *CHES*, volume 2162 of *Lecture Notes in Computer Science*, pages 309–318. Springer, 2001.

[5] P. Barreto and V. Rijmen. The KHAZAD Legacy-Level Block Cipher. Submission to NESSIE project, available from http://www.cosic.esat.kuleuven.ac.be/nessie/, 2000.

[6] Lejla Batina, Kerstin Lemke, Elisabeth Oswald, Gilles Piret, and François-Xavier Standaert. Electromagnetic Analysis and Fault Attacks: State of the Art. Deliverable D.VAM.4, European Network of Excellence in Cryptology, ECRYPT, 2005.

[7] Jürgen Becker, Marco Platzner, and Serge Vernalde, editors. *Field Programmable Logic and Application, 14th International Conference , FPL 2004, Leuven, Belgium, August 30-September 1, 2004, Proceedings*, volume 3203 of *Lecture Notes in Computer Science*. Springer, 2004.

[8] Eli Biham, Ross J. Anderson, and Lars R. Knudsen. Serpent: A New Block Cipher Proposal. In Serge Vaudenay, editor, *Fast Software Encryption*, volume 1372 of *Lecture Notes in Computer Science*, pages 222–238. Springer, 1998.

[9] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation Power Analysis with a Leakage Model. In Joye and Quisquater [17], pages 16–29.

[10] Çetin Kaya Koç and Christof Paar, editors. *Cryptographic Hardware and Embedded Systems - CHES 2000, Second International Workshop, Worcester, MA, USA, August 17-18, 2000, Proceedings*, volume 1965 of *Lecture Notes in Computer Science*. Springer, 2000.

[11] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards Sound Approaches to Counteract Power-Analysis Attacks. In Wiener [54], pages 398–412.

[12] Christophe Clavier, Jean-Sébastien Coron, and Nora Dabbous. Differential Power Analysis in the Presence of Hardware Countermeasures. In Çetin Kaya Koç and Paar [10], pages 252–263.

[13] Jean-Sébastien Coron, Paul C. Kocher, and David Naccache. Statistics and Secret Leakage. In Yair Frankel, editor, *Financial Cryptography*, volume 1962 of *Lecture Notes in Computer Science*, pages 157–173. Springer, 2000.

[14] Louis Goubin and Jacques Patarin. DES and Differential Power Analysis (The "Duplication" Method). In Çetin Kaya Koç and Christof Paar, editors, *CHES*, volume 1717 of *Lecture Notes in Computer Science*, pages 158–172. Springer, 1999.

[15] A. Hald. *Statistical Theory with Engineering Applications.* John Wiley & Sons, Inc., 1952.

[16] ISO/IEC. IEC 61967-4: Integrated circuits - Measurement of electromagnetic emissions, 150 kHz to 1 GHz - Part 4: Measurement of conducted emissions 1Ω / 150Ω. Direct coupling method, 47A/636/FDIS, January 2002.

[17] Marc Joye and Jean-Jacques Quisquater, editors. *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, volume 3156 of *Lecture Notes in Computer Science.* Springer, 2004.

[18] Chris Karlof and David Wagner. Hidden Markov Model Cryptoanalysis. In Walter et al. [52], pages 17–34.

[19] Steven M. Kay. *Fundamentals of Statistical Signal Processing, Volume 1 and 2.* Prentice-Hall, Inc., 1993.

[20] N. Koblitz. Elliptic curve cryptosystem. *Math. Comp.*, 48:203–209, 1987.

[21] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In Michael Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.

[22] François Mace, François-Xavier Standaert, Ilham Hassoune, Jean-Didier Legat, and Jean-Jacques Quisquater. A Dynamic Current Mode Logic to Counteract Power Analysis Attacks. In *DCIS*, pages 186–191, 2004.

[23] Stefan Mangard. Hardware Countermeasures against DPA? A Statistical Analysis of Their Effectiveness. In Tatsuaki Okamoto, editor, *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 222–235. Springer, 2004.

[24] Stefan Mangard, Thomas Popp, and Berndt M. Gammel. Side-Channel Leakage of Masked CMOS Gates. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 351–365. Springer, 2005.

[25] M. M. Mano and C. R. Kime. *Logic and Computer Design Fundamentals.* Prentice Hall, Upper Saddle River, New Jersey 07458, second edition, 2001.

[26] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, oktober 1996.

[27] Thomas S. Messerges. Using Second-Order Power Analysis to Attack DPA Resistant Software. In Çetin Kaya Koç and Paar [10], pages 238–251.

[28] Thomas S. Messerges, Ezzy A. Dabbish, and Robert H. Sloan. Investigations of Power Analysis Attacks on Smartcards. In *USENIX Workshop on Smartcard Technology (Smartcard '99)*, pages 151–162, May 1999.

[29] Thomas S. Messerges, Ezzy A. Dabbish, and Robert H. Sloan. Power Analysis Attacks of Modular Exponentiation in Smartcards. In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES'99, First International Workshop, Worcester, MA, USA, August 12-13, 1999, Proceedings*, volume 1717 of *Lecture Notes in Computer Science*, pages 144–157. Springer, 1999.

[30] V. Miller. Uses of elliptic curves in cryptography. In H. C. Williams, editor, *Advances in Cryptology: Proceedings of CRYPTO'85*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426, Santa Barbara, CA, USA, August 18-22 1985. Springer-Verlag.

[31] National Bureau of Standards. FIPS PUB 46, The Data Encryption Standard. Federal Information Processing Standard, NIST, U.S. Dept. of Commerce., 1977.

[32] National Bureau of Standards. FIPS 197, Advanced Encryption Standard. Federal Information Processing Standard, NIST, U.S. Dept. of Commerce., 2001.

[33] Siddika Berna Örs, Frank K. Gürkaynak, Elisabeth Oswald, and Bart Preneel. Power-Analysis Attack on an ASIC AES implementation. In *ITCC (2)*, pages 546–552. IEEE Computer Society, 2004.

[34] Siddika Berna Örs, Elisabeth Oswald, and Bart Preneel. Power-Analysis Attacks on an FPGA - First Experimental Results. In Walter et al. [52], pages 35–50.

[35] E. Oswald, S. Mangard, and N. Pramstaller. Secure and Efficient Masking of AES - A Mission Impossible? IACR e-print archive 2004/134, http://eprint.iacr.org, 2004.

[36] Elisabeth Oswald, Stefan Mangard, Norbert Pramstaller, and Vincent Rijmen. A Side-Channel Analysis Resistant Description of the AES S-Box. In Henri Gilbert and Helena Handschuh, editors, *FSE*, volume 3557 of *Lecture Notes in Computer Science*, pages 413–423. Springer, 2005.

[37] Eric Peeters, François-Xavier Standaert, Nicolas Donckers, and Jean-Jacques Quisquater. Improved Higher-Order Side-Channel Attacks with FPGA experiments. In Berk Sunar and Josyula R. Rao, editors, *CHES*, Lecture Notes in Computer Science. Springer, 2005.

[38] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[39] L. Shang, A. S. Kaviani, and K. Bathala. Dynamic Power Consumption in Virtex-II FPGA family. In *Proceedings of the 2002 ACM/SIGDA 10th International Symposium on Field-Programmable Gate Arrays*, pages 157–164. ACM Press, 2002.

[40] François-Xavier Standaert, François Mace, Eric Peeters, and Jean-Jacques Quisquater. Updates on the Security of FPGAs against Power Analysis Attacks. In *Applied Reconfigurable Computing (ARC)*, 2006.

[41] François-Xavier Standaert, Siddika Berna Örs, and Bart Preneel. Power Analysis of an FPGA: Implementation of Rijndael: Is Pipelining a DPA Countermeasure? In Joye and Quisquater [17], pages 30–44.

[42] François-Xavier Standaert, Siddika Berna Örs, Jean-Jacques Quisquater, and Bart Preneel. Power Analysis Attacks Against FPGA Implementations of the DES. In Becker et al. [7], pages 84–94.

[43] François-Xavier Standaert, Eric Peeters, and Jean-Jacques Quisquater. On the Masking Countermeasure and Higher-Order Power Analysis Attacks. In *ITCC (1)*, pages 562–567. IEEE Computer Society, 2005.

[44] François-Xavier Standaert, Eric Peeters, Gaël Rouvroy, and Jean-Jacques Quisquater. An Overview of Power Analysis Attacks Against Field Programmable Gate Arrays. In *Proceedings of the IEEE*, 2006.

[45] Kris Tiri, M. Akmal, and Ingrid Verbauwhede. A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards. In *ESSCIRC*, pages 403–406, 2002.

[46] Kris Tiri and Ingrid Verbauwhede. Securing Encryption Algorithms against DPA at the Logic Level: Next Generation Smart Card Technology. In Walter et al. [52], pages 125–136.

[47] Kris Tiri and Ingrid Verbauwhede. A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation. In *DATE*, pages 246–251. IEEE Computer Society, 2004.

[48] Kris Tiri and Ingrid Verbauwhede. Secure Logic Synthesis. In Becker et al. [7], pages 1052–1056.

[49] Elena Trichina. Combinational Logic Design for AES SubByte Transformation on Masked Data. Cryptology ePrint Archive, Report 2003/236, 2003. `http://eprint.iacr.org/`.

[50] Emmanuelle Dottax Vincent Carlier, Hervé Chabanne and Hervé Pelletier. Electromagnetic Side Channels of an FPGA Implementation of AES. Cryptology ePrint Archive, Report 2004/145, 2004. `http://eprint.iacr.org/`.

[51] Jason Waddle and David Wagner. Towards Efficient Second-Order Power Analysis. In Joye and Quisquater [17], pages 1–15.

[52] Colin D. Walter, Çetin Kaya Koç, and Christof Paar, editors. *Cryptographic Hardware and Embedded Systems - CHES 2003, 5th International Workshop, Cologne, Germany, September 8-10, 2003, Proceedings*, volume 2779 of *Lecture Notes in Computer Science*. Springer, 2003.

[53] N. Weste and K. Eshraghian. *Principles of CMOS VLSI Design*. Addison-Wesley, 2nd edition, 1993.

[54] Michael J. Wiener, editor. *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*. Springer, 1999.

[55] Thomas J. Wollinger and Christof Paar. How Secure Are FPGAs in Cryptographic Applications? In Peter Y. K. Cheung, George A. Constantinides, and José T. de Sousa, editors, *FPL*, volume 2778 of *Lecture Notes in Computer Science*, pages 91–100. Springer, 2003.

[56] Xess Corporation. XSV Board V1.1 Manual. `http://www.xess.com/manuals/xsv-manual-v1_1.pdf`, September 2001.

[57] Xilinx. Virtex 2.5 V Field Programmable Gate Arrays. `http://direct.xilinx.com/bvdocs/publications/ds003.pdf`, April 2001.

[58] Xilinx. Powering Xilinx FPGAs. `http://www.xilinx.com/xapp/xapp158.pdf`, August 2002.

[59] Xilinx, Inc. *Virtex 2.5 V Field Programmable Gate Arrays*, April 2 2001. `http://direct.xilinx.com/bvdocs/publications/ds083.pdf`.