Power-Analysis Attacks on an FPGA – First Experimental Results

Sıddıka Berna Örs[1], Elisabeth Oswald[2,3], and Bart Preneel[1]

[1] Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC, Kasteelpark Arenberg 10, B–3001 Leuven-Heverlee, Belgium
[2] Institute for Applied Information Processing and Communications (IAIK), TU Graz, Inffeldgasse 16a, A–8010 Graz, Austria
[3] A–SIT, Technologiebeobachtung, Inffeldgasse 16a, A–8010 Graz, Austria
{siddika.bernaors, bart.preneel}@esat.kuleuven.ac.be
elisabeth.oswald@iaik.at †

Abstract. Field Programmable Gate Arrays (FPGAs) are becoming increasingly popular, especially for rapid prototyping. For implementations of cryptographic algorithms, not only the speed and the size of the circuit are important, but also their security against implementation attacks such as side-channel attacks. Power-analysis attacks are typical examples of side-channel attacks, that have been demonstrated to be effective against implementations without special countermeasures. The flexibility of FPGAs is an important advantage in real applications but also in lab environments. It is therefore natural to use FPGAs to assess the vulnerability of hardware implementations to power-analysis attacks. To our knowledge, this paper is the first to describe a setup to conduct power-analysis attacks on FPGAs. We discuss the design of our hand-made FPGA-board and we provide a first characterization of the power consumption of a Virtex 800 FPGA. Finally we provide strong evidence that implementations of elliptic curve cryptosystems without specific countermeasures are indeed vulnerable to simple power-analysis attacks.
Keywords: FPGA, Power Analysis, Elliptic Curve Cryptosystems

1 Introduction

Since their publication in 1998, power-analysis attacks have attracted significant attention within the cryptographic community. So far, they have been successfully applied to different kinds of (unprotected) implementations of symmetric

† Sıddıka Berna Örs is funded by a research grant of the Batholites Universiteit Leuven, Belgium. Dr. Bart Preneel is professor at the Batholites Universiteit Leuven. Elisabeth Oswald is with the TU Graz and the A–SIT. This work was supported by Concerted Research Action GOA-MEFISTO-666 of the Flemish Government, by the FWO "Identification and Cryptography" project (G.0141.03) and by the FWF "Investigations of Simple and Differential Power Analysis" project (P16110-N04).

and public-key encryption schemes and on digital signature schemes. Most attacks which have been published in the open literature apply to software implementations of cryptographic algorithms which can be found for example in smart cards (see [KJJ99], [MDS99a] or [MDS99b]). However, modern smart cards and accelerators for cryptographic algorithms also contain hardware implementations of cryptographic algorithms.

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 examples) for a circuit can easily be ported, if not directly used, for an FPGA implementation of the circuit. Naturally, it is desirable to use the resulting FPGA implementation also for an evaluation of the designed circuit against power-analysis attacks.

This article describes the first realization of power-analysis attacks on a Virtex FPGA. We can prove that this FPGA leaks a significant amount of information about its internal computations through the supply lines. We can even provide evidence that the power consumption characteristics are comparable with the power consumption characteristics of ordinary ASICs. To demonstrate how dramatic the power consumption leakage of this FPGA is, we finally perform a simple power-analysis attack on an implementation of an elliptic-curve point-multiplication.

The remainder of this article is organized as follows. We recall the principles of power-analysis attacks in Sect. 2. FPGAs are introduced in Sect. 3. For the purpose of conducting power-analysis attacks, we built a special measurement board. This measurement setup is described in Sect. 4. The results of our experiments can be found in Sect. 5. Section 6 presents the conclusion of our research.

1.1 Related Work

The characterization of the power-consumption characteristics of FPGAs has received little attention so far. Shang *et al.* [SKB02] is the only recent article in that field. In their article, Shang *et al.* 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. Based on this result, it seems much more difficult to conduct power-analysis attacks on FPGAs than on ASICs. However, as we will demonstrate in this article, such attacks are feasible and can be realized in practise.

2 Power-Analysis Attacks

Power-analysis attacks are a very powerful type of side-channel attack, published first by Kocher *et al.* [KJJ99]. Power-analysis attacks are passive in the sense that an attacker only needs to measure the power consumption of a device without manipulating it actively, that is, an attacker uses the device in its intended mode of use. A likely scenario (in the case of attacks on smart cards) is that an attacker lets the device execute an internal authenticate command. While the device is executing this command, the attacker measures the power consumed by the device. Statistical methods allow to extract efficiently the information on the secret key that is contained in the measurements.

2.1 Power Consumption Characteristics of CMOS

Nowadays, almost all smart card processors are implemented in CMOS (complementary Metal-Oxid Silicon) technology. In CMOS technology, the values 0 and 1 are represented by V_{ss} and V_{dd}, respectively. The dominating factor for the power consumption of a CMOS gate is the dynamic power consumption [WE93]. *Transition count* leakage and *Hamming weight* leakage can typically be observed in CMOS circuits, see [MDS99b] 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 switching of the gates causes a current flow which is visible through both V_{dd} and V_{ss}. This current flow can be observed on the outside of the device by (for example) putting a small resistor between the devices V_{ss} or V_{dd} and the true V_{dd}. The current flowing through the resistor creates a voltage which can be measured by a digital oscilloscope.

2.2 Exploiting the (Hidden) Information

Depending on how direct the information about the power consumption can be used, simple or differential power-analysis attacks (SPA or DPA) [KJJ99] have to be applied. SPA attacks are always possible when the power consumption is more or less directly related to the actions of the secret key. This is mostly the case when the instructions executed in the device give evidence about the secret key. If the instructions do not provide such information, but the processed data do so instead, then the information is typically more hidden in the overall power consumption and thus statistical methods have to be applied to bring them to light. This is the approach taken for differential power-analysis attacks.

Simple Power-Analysis Attacks Simple power-analysis attacks exploit the relationship between the instantaneous power consumption of a device and the instructions that are executed. For simple power-analysis attacks it is assumed that every instruction has its unique power-consumption trace. An attacker simply monitors the device's power consumption while it performs a cryptographic operation. Then, the attacker carefully studies the obtained power-consumption trace to determine the sequence of instructions performed by the device. If this sequence is directly related to the secret key which was involved in the cryptographic operation, the attacker can deduce this secret key from the power-consumption trace. Such an attack typically targets implementations which use key dependent branching in the implementation.

Differential Power-Analysis Attacks An attacker faces now the task of exploiting the hidden information about the secret key in an efficient way. For this purpose, the attacker creates a hypothetical model of the device. This hypothetical model describes, at a very abstract level, the instantaneous power consumption of the device when it executes a certain cryptographic algorithm. For this purpose, at least a small part of the unknown key has to be guessed. Fortunately (for the attacker), all algorithms deployed in practice use only small parts of the secret key at a time.

The attacker writes a simple computer program that executes the algorithm (or at least a small part of it, where a part of the key is used). The program calculates the result (of this part) for all possible key values. These values allow to predict the power consumption, which is for example related to the Hamming-weight of the internal data.

In the last stage of the attack, an attacker feeds the same input values which he used in the model to the real device and measures its power consumption. Then the attacker correlates the predictions of the model with the real power consumption values. For all the wrong key guesses, the predictions will not correlate with the real measurements, but for the correct key guess, there will be a peak visible in the correlation trace.

3 Field Programmable Logic Arrays

An FPGA consists of an array of configurable logic blocks (CLBs), surrounded by programmable I/O blocks, and connected with programmable interconnections as shown in Fig. 1 [Opt]. A typical FPGA contains from 64 to tens of thousands of logic blocks and an even greater number of flip-flops. Most FPGAs do not provide a 100% interconnect between the logic blocks. Instead, sophisticated software places and routes the logic on the device.

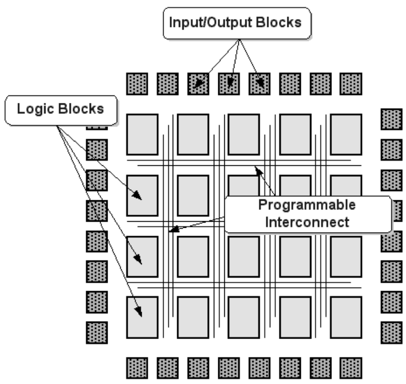

Fig. 1. The FPGA architecture

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 has benefits for more plentiful programmable interconnect.

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 [MK01].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. More details on the Xilinx Virtex Architecture are provided in Appendix A.

4 The Measurement Setup

Our setup consists of essentially two boards (see Fig. 2). The main board is responsible for interfacing the PC via the parallel port. It is connected with the XILINX parallel cable in order to program the VIRTEX FPGA and it provides some LEDs, switches and buttons for testing purposes. The daughter board itself just carries the VIRTEX FPGA, it allows to access some pins for triggering and to measure the power consumption of the VIRTEX FPGA in a convenient way.

4.1 The Mother Board

The Parallel Port [Axe00] is the most commonly used port for interfacing home made projects. This port allows the input of 5 bits and the output of 12 bits. The port is composed of 4 control lines, 5 status lines and 8 data lines. The communication between the FPGA and the PC uses this parallel port. We need only 17 input/output pins to send data or commands to the FPGA and receive the result, but we designed the board in such a way that it gives us more monitoring points and thus connected 32 input/output pins of the FPGA to the board. The unused input/output pins are pulled up after the configuration.

We also designed a protocol to send and receive data to and from 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 status lines are used for sending the result from the FPGA to the PC. The protocol is independent from the operation executed in the FPGA. Only the length of

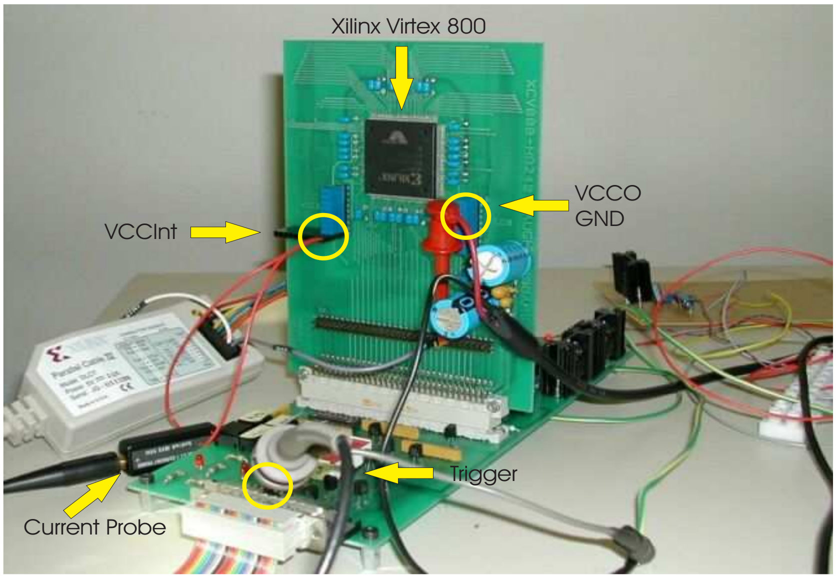

Fig. 2. The measurement setup. On the daughter board the current probe is connected to VCCINT. Alternatively it can be connected to the VCCO of the individual banks, or the GND.

the data which is communicated can be modified by the PC. This provides a flexible setup where experiments with different algorithms can be performed in a coherent manner.

4.2 The Daughter Board

We use a Xilinx XCV800 FPGA from the Virtex series in a HQ240C package. Reasons for this particular choice include:

1. The resources are sufficient to implement a 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 should draw more current from their supply lines. With our setup it is possible to verify this hypothesis. In case that different parts of a design (such as an elliptic-curve addition and an elliptic-curve doubling) are mapped to different banks of the FPGA, measuring the current of the individual banks allows us 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 Fig. 2. During the normal operation of the board without measurement the two pins are connected by a jumper. When we want 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 Fig. 2.

This setup gives the possibility of making measurements on different points at the same time and makes it easy to modify the measurement point.

Bypassing Considerations With high-speed, high-density FPGA devices, maintaining signal integrity is the key to reliable, repeatable designs [Xil02]. Proper power bypassing and decoupling improves the overall signal integrity. Without it, power and ground voltages are affected by logic transitions and can cause operational issues.

When a logic device switches from a logic one to a logic zero, or a logic zero to a logic one, the output structure is momentarily at a low impedance across the power supply. Each transition requires that a signal line be charged or discharged, which requires energy. As a result, many electrons are suddenly needed to keep the voltage from collapsing. The function of the bypass capacitor is to provide local energy storage.

$10nF$ capacitors are placed between every VCCINT and VCCO pin of the FPGA and the nearest GND. Because we designed the setup in two different cards, the daughter card can be thought as a stand alone chip taking power from the mother board. Bypass capacitors had to be placed between the power supplies of the card and the GND line.

5 Results

We now describe the experiments conducted on the measurement setup described above. As the aim of our work was to build an alternative platform for power-analysis attacks, we decided to perform first some basic experiments to verify that the assumption which we usually make for such attacks are also valid for our FPGA setup.

5.1 Power Consumption Characteristics

As discussed in Sect. 2, we should be able to detect either transition-count leakage or Hamming-weight leakage in our setup. This is because according to Sect. 3, the CLBs consist of flip-flops (and other logic) which exhibit the power consumption characteristics of CMOS technology. The only problem can be that if the circuit, which we load into the FPGA, does not use all of the FPGAs resources, then the noise which is produced by the unused parts might be larger than the signal produced by the circuit.

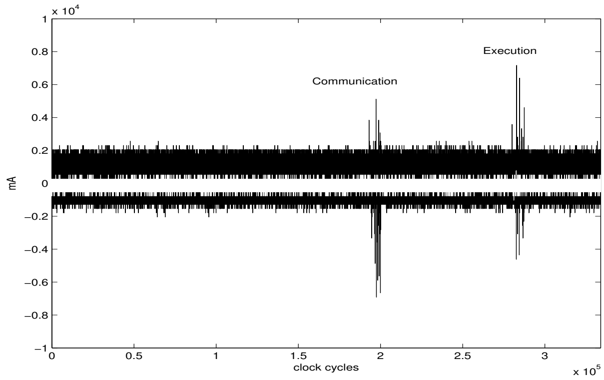

Fig. 3. Comparison between an idle bank, which corresponds to the white trace in this picture, and a bank which receives data (8 bits) and processes it, which corresponds to the dark trace in this picture.

To evaluate the behavior of the FPGA we loaded a small circuit on one of the banks of the FPGA. Then, we measured the power consumption of the whole FPGA and, at the same time, the power consumption of an empty (idle) bank (see Fig. 3 for the power consumption traces). The overall power consumption (the dark trace) shows clearly peaks when data is transmitted and when the data is processed. The light trace however does not exhibit any peaks during the whole computation. This experiment confirms that idle parts of the FPGA will not influence the overall power consumption. Moreover, even the power consumption of a very small circuit (we used only 3% of the FPGA and of the FPGAs flip-flops) can be easily detected.

With another simple set of experiments we confirmed that the amount of power consumed of the FPGA is linear in the number of switched flip-flops. We have designed registers of a specific size and loaded them on the FPGA. Then we let them repeatedly store 0 and 1 value and measured the FPGA's power consumption. Fig. 4 and 5 illustrate that the power consumed by the 6000-bit register for storing all 1s is about twice as high as the power consumed by the 3000-bit register.

A direct conclusion from such experiments is that the power consumption characteristics are essentially the same as of an ordinary CMOS circuit. Idle CLBs or even idle banks do not add too much noise to the overall power consumption.

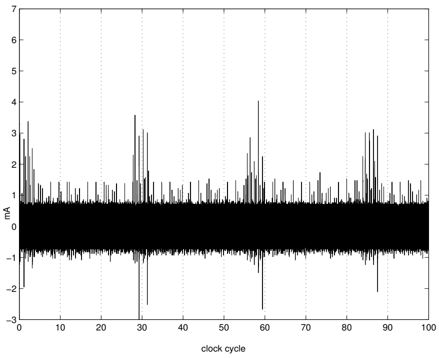

Fig. 4. Power consumption trace of a 3000-bit register.

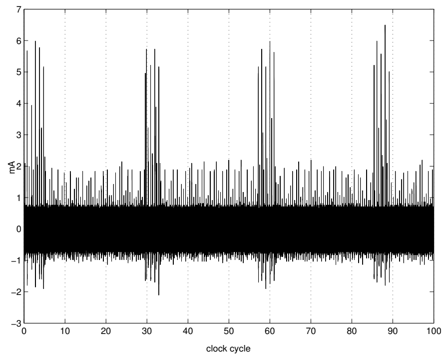

Fig. 5. Power consumption trace of a 6000-bit register.

5.2 Attacking an Implementation of an Elliptic-Curve Point-Multiplication

With the experience gained from these experiments, we attacked an implementation of an EC point multiplication. We have implemented the arithmetic for a 160-bit prime field with a Montgomery modular multiplier (MMM) without final subtraction ([Mon85],[ÖBPV03], see Algorithm 1 for a description).

Algorithm 1 Montgomery modular multiplication without final subtraction

Input: Integers $N = (n_{l-1} \cdots n_1 n_0)_2$, $x = (x_l \cdots x_1 x_0)_2$, $y = (y_l \cdots y_1 y_0)_2$ with $x \in [0, 2N-1], y \in [0, 2N-1]$, $R = 2^{l+2}$, $gcd(N, 2) = 1$ and $N' = -N^{-1} \mod 2$ (Notation $T = (t_l t_{l-1} ... t_0)$)

Output: $xyR^{-1} \mod 2N$

$T \leftarrow 0$
for i from 0 to $l+1$ **do**
  $m_i \leftarrow (t_0 + x_i y_0)\, N' \bmod 2$
  $T \leftarrow (T + x_i y + m_i N)/2$
end for
Return (T)

To obtain a linear, pipelined modular multiplier, a systolic array shown in Fig. 6 is used [Wal99]. $X(0)$ denotes the least significant bit (LSB) of the register in which the input x is stored. T denotes the intermediate value register. The carry chain is stored in the $C0$ and $C1$ registers. The Montgomery modular multiplication circuit (MMMC) consists of a controller and a data path. The data path consists of a systolic array, four internal registers, a counter and a comparator.

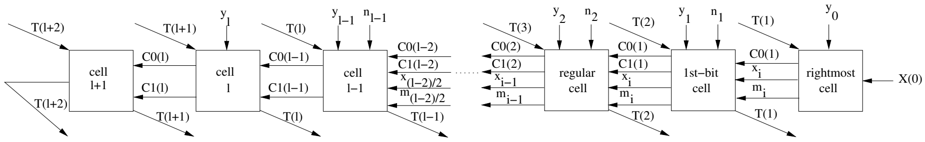

Fig. 6. Schematic view of the complete systolic array

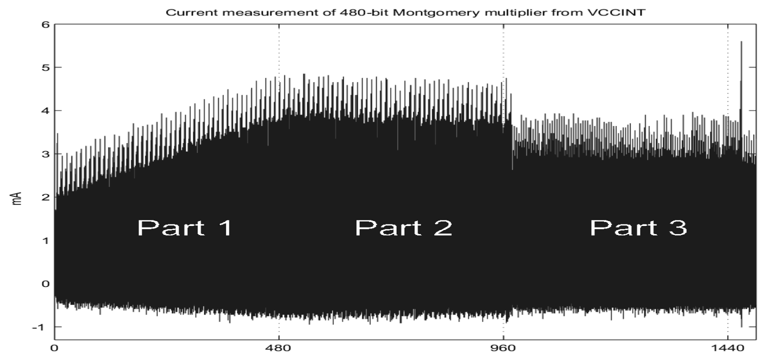

Fig. 7. Power consumption trace of 480-bit Montgomery modular multiplier from VC-CINT

The measurement of a 480-bit MMMC is depicted in Fig. 7. The three parts shown in the figure can be explained according to the algorithm and architecture used. The T register in Fig. 6 is reset in the beginning of the MMM operation and then it is being written. The number of bits in T which are updated is increasing until clock cycle l. This stage corresponds to the first part shown in power consumption trace. After l clock cycles all the bits of the T register have a value and all of them are updated before clock cycle $2l$. This stage is shown by the second part in Fig. 7. The last part in Fig. 7 corresponds to reading out the result from the pipeline. Because there is no new input on the LSB of the systolic array, starting from clock cycle $2l + 1$ the number of MSBs of the T register that are updated decreases.

5.3 Elliptic Curve Point Addition and Doubling

For the representation of the points on the elliptic curve we use modified Jacobian coordinates as proposed by Cohen *et al.* in [CMO98]. These points are represented as quadruple $\left(X, Y, Z, aZ^4\right)$. When we convert the input point P from affine coordinates to projective coordinates we take Z as 1. Because there are both MMMC and modular addition/subtraction (MAS) circuits available, these operations can be executed in parallel. When an EC point addition is used in an EC point multiplication one of the inputs of the EC point addition circuit is always the input point P. Algorithm 2.(a). and (b)., describe the point addition and the point doubling operation, resp.

Algorithm 2 EC point addition and doubling

(a)	(b)
Input: $P_1 = (x, y, 1, a)$, $P_2 = (X_2, Y_2, Z_2, aZ_2^4)$	**Input:** $P_1 = (X_1, Y_1, Z_1, aZ_1^4)$
Output: $P_1 + P_2 = P_3 = (X_3, Y_3, Z_3, aZ_3^4)$	**Output:** $2P_1 = P_3 = (X_3, Y_3, Z_3, aZ_3^4)$
1. $T_1 \leftarrow Z_2^2$	1. $T_1 \leftarrow Y_1^2$, $T_2 \leftarrow 2X_1$
2. $T_2 \leftarrow xT_1$	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. $T_1 \leftarrow yT_1$	4. $T_2 \leftarrow X_1^2$, $T_3 \leftarrow 2T_3$
5. $T_4 \leftarrow T_3^2$, $T_5 \leftarrow Y_2 - T_1$	5. $T_4 \leftarrow Y_1Z_1$, $T_3 \leftarrow 2T_3$
6. $T_2 \leftarrow T_2T_4$,	6. $T_5 \leftarrow T_3\left(aZ_1^4\right)$, $T_6 \leftarrow 2T_2$
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. $T_2 \leftarrow T_2 + \left(aZ_1^4\right)$
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. $Y_3 \leftarrow T_2 - T_3$
(a)	**(b)**

The multiplications and the squarings use MMMC, while the additions, doublings and subtractions employ the modular addition/subtraction circuit. The power consumption trace of one 160-bit EC point addition is shown in Fig. 9. Fourteen states can be counted easily from the trace. All the states are completed in nearly 500 clock cycles The power consumption during Step 3, 5, 7, 8, 10, 11 and 13 seems higher than for the other steps. In these steps a modular addition or subtraction is taking place as well as an MMM.

The MAS operation is performed in two steps as addition-subtraction or subtraction-addition. Depending on the result of the first operation the second operation takes place or is ignored. This behaviour can be observed when we zoom in Step 3 as shown in Fig. 8. This figure shows that after 160 cycles the first subtraction ends and the next addition operation start. The addition lasts 160 clock cycles.

The power consumption trace of one 160-bit EC point doubling is shown in Fig. 10. As expected, the number of clock cycles for EC point doubling is less than the number of clock cycles for EC point addition. The main difference in power consumption between EC point addition and EC point doubling can be observed by looking at Step 7, 8, 10, 11, 12 and 14. In these steps only modular a addition/subtraction takes place. Obviously the latency and power consumption of these are smaller than the others. This means that a simple power-analysis attack is easy to perform.

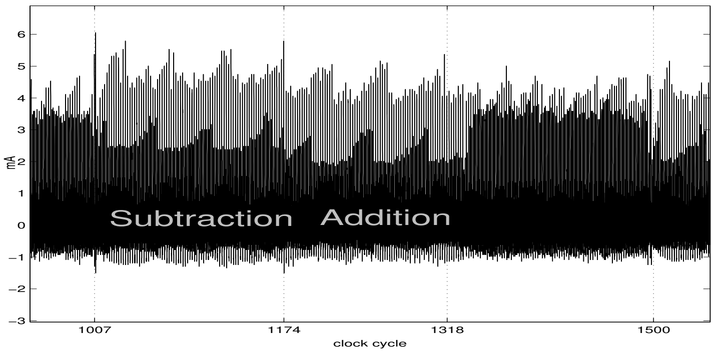

Fig. 8. Power consumption trace of Step 3 of 160-bit EC point addition

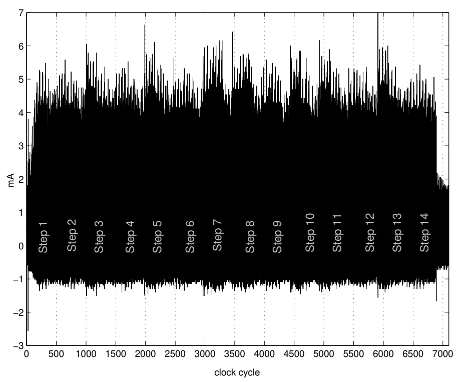

Fig. 9. Power consumption trace of 160-bit EC point addition from VCCINT

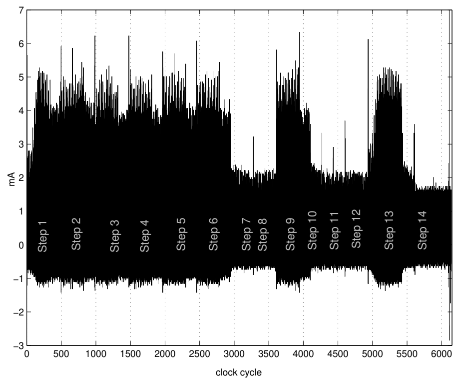

Fig. 10. Power consumption trace of 160-bit EC point doubling from VCCINT

The EC point multiplication is implemented by using a simple double-and-add algorithm. For EC point addition and EC point doubling the circuits described above are used. The power consumption trace of a 160-bit EC point multiplication is shown in Fig 11. It can be easily seen from figure 11 that the key used during this measurement is 1001100.

5.4 Applications and Future Work

Our board makes it possible to verify the effectiveness of many of the proposed countermeasures for various algorithms. In particular, we believe that countermeasures that are based on masking or blinding intermediate values (see for example [Koc96] and [Cor99] for approaches on asymmetric schemes), can be evaluated with our board. Also countermeasures for elliptic curve cryptosystems

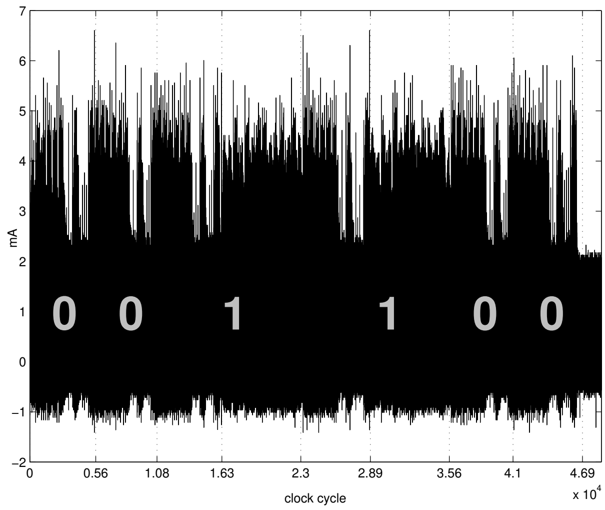

Fig. 11. Power consumption trace of a 160-bit EC point multiplication from VCCINT.

which are based on clever implementations of the elliptic curve operations [TB03] can be checked. All software based countermeasures can be evaluated with our setup. We plan to validate some of the countermeasures, such as [TB03] and to apply EM attacks [GMO01] on the FPGA.

6 Conclusion

We introduced a new platform for evaluating power analysis. Our approach consists of an FPGA, which is placed on a hand-made board which makes it very easy to conduct power-analysis attacks. We characterized the power consumption of a XILINX Virtex 800 FPGA and conclude that it is similar to the power consumption of an ordinary ASIC in CMOS technology. Therefore, it is possible to draw conclusions about the vulnerability of a certain circuit by performing power-analysis attacks on an FPGA-implementation. Since programming an FPGA is considerably cheaper than manufacturing an ASIC, assessing a devices vulnerability towards power-analysis attacks is much cheaper on our platform. Consequently, our approach describes the first cheap and efficient way to conduct power-analysis attacks on a real implementation (i.e., not on a software simulation) of a circuit in a very early stage of the design flow.

References

[Axe00] J. Axelson. *Parallel Port Complete: Programming, Interfacing, and Using the PC's Parallel Printer Port.* Lakeview Research, Madison, WI 53704, 2000.

[CMO98] H. Cohen, A. Miyaji, and T. Ono. Efficient elliptic curve exponentiation using mixed coordinates. In K. Ohta and D. Pei, editors, *Proceedings of ASIACRYPT 1998*, number 1514 in Lecture Notes in Computer Science, pages 51–65. Springer-Verlag, 1998.

[Cor99] J.-S. Coron. Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, August 12-13, 1999, Proceedings*, volume 1717 of *Lecture Notes in Computer Science*, pages 292–302. Springer, 1999.

[GMO01] K. Gandolfi, Ch. Mourtel, and F. Olivier. Electromagnetic Analysis: Concrete Results. In Çetin Kaya Koç, David Naccache, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings*, volume 2162 of *Lecture Notes in Computer Science*, pages 251–261. Springer Verlag, 2001.

[KJJ99] P. C. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis. In Michael Wiener, editor, *Advances in Cryptology-CRYPTO 1999*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.

[Koc96] P. C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *Proceedings of Crypto'96*, number 1109 in Lecture Notes in Computer Science, pages 104–113. Springer, 1996.

[MDS99a] T. S. Messerges, E. A. Dabbish, and R. H. Sloan. Power Analysis Attacks of Modular Exponentiation in Smartcards. In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, August 12-13, 1999, Proceedings*, volume 1717 of *Lecture Notes in Computer Science*, pages 144–157. Springer, 1999.

[MDS99b] T. S. Messerges, E. A. Dabbish, and R.H. Sloan. Investigations of Power Analysis Attacks on Smartcards. In *Proceedings of USENIX Workshop on Smartcard Technology*, pages 151–162, 1999.

[MK01] M. M. Mano and C. R. Kime. *Logic and Computer Design Fundamentals.* Prentice Hall, Upper Saddle River, New Jersey 07458, second edition, 2001.

[Mon85] P. Montgomery. Modular multiplication without trial division. *Mathematics of Computation*, Vol. 44:519–521, 1985.

[ÖBPV03] S. B. Örs, L. Batina, B. Preneel, and J. Vandewalle. Hardware implementation of an elliptic curve processor over GF(p). In *The 10th Reconfigurable Architectures Workshop (RAW)*, Nice, France, April 2003.

[Opt] OptiMagic, Inc. Frequently-Asked Questions (FAQ) About Programmable Logic. http://www.optimagic.com/faq.html#FPGA.

[SKB02] 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.

[TB03] E. Trichina and A. Bellezza. Implementation of Elliptic Curve Cryptography with built-in Countermeasures against Side Channel Attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, volume 2535 of *Lecture Notes in Computer Science (LNCS)*, pages 98–113. Springer, 2003.

[Wal99] C. D. Walter. Montgomery's multiplication technique: How to make it smaller and faster. In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, August 12-13, 1999, Proceedings*, volume 1717 of *Lecture Notes in Computer Science*, pages 80–93. Springer, 1999.

[WE93] N. Weste and K. Eshraghian. *Principles of CMOS VLSI Design*. Addison-Wesley, 2nd edition, 1993.

[Xil01] Xilinx, Inc. *Virtex 2.5 V Field Programmable Gate Arrays*, April 2 2001. http://direct.xilinx.com/bvdocs/publications/ds083.pdf.

[Xil02] Xilinx, Inc. *Powering Xilinx FPGAs*, August 5 2002. http://support.xilinx.com/xapp/xapp158.pdf.

A 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.

Configurable Logic Block The basic building block of the Virtex CLB is the logic cell (LC) [Xil01]. A LC includes a 4-input function generator, carry logic, and a storage element. The output from the function generator in each LC drives both the CLB output and the D input of the flip-flop. Each Virtex CLB contains four LCs, organized in two similar slices. Figure 12 shows a more detailed view of a single slice. In addition to the four basic LCs, 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 storage elements in the Virtex slice can be configured either as edge-triggered D-type flip-flops or as level-sensitive latches. The D inputs can be driven either by the function generators within the slice or directly from the slice inputs, bypassing the function generators. In addition to Clock and Clock Enable signals, each Slice has synchronous set and reset signals (SR and BY). All the control signals can be inverted independently and are shared by the two flip-flops within the slice.

I/O Block The Virtex I/O Block (IOB) features SelectIO inputs and outputs that support a wide variety of I/O signalling standards [Xil01]. The three IOB storage elements function either as edge-triggered D-type flip-flops or as level sensitive latches. Optional pull-up and pull-down resistors and an optional weak-keeper circuit are attached to each pad. Prior to configuration, all pins not involved in configuration are forced into their high-impedance state.

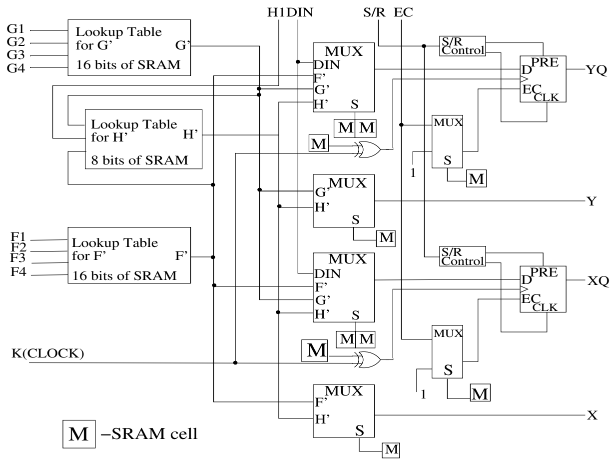

Fig. 12. Simplified diagram

I/O Banking Some of the possible I/O standards require VCCO and/or VREF 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 (Output supply voltage) pins, all of which must be connected to the same voltage. This voltage is determined by the output standards in use.

Configuration of the FPGA Virtex devices are configured by loading configuration data into the internal configuration memory. Some of the pins used for this are dedicated configuration pins, while others can be re-used as general purpose inputs and outputs once configuration is complete. Virtex supports four configuration modes which are the Slave-serial mode, the Master-serial mode, the SelectMAP mode and the Boundary-scan mode. The configuration mode pins (M2, M1, and M0) define which of these modes is used. Our board supports three of these configuration modes.