Smartly Analyzing the Simplicity and the Power of Simple Power Analysis on Smartcards

Rita Mayer-Sommer

Electrical Engineering Division, ETH Zürich, Switzerland
rmayerso@ee.ethz.ch

Abstract. A new kind of cryptanalytic attacks, targeted directly at the weaknesses of a cryptographic algorithm's physical implementation, has recently attracted great attention. Examples are timing, glitch, or power-analysis attacks. Whereas in so-called *simple power analysis* (SPA for short) only the power consumption of the device is analyzed, *differential power analysis* (DPA) additionally requires knowledge of ciphertext outputs and is thus more costly. Previous investigations have indicated that SPA is little threatening and moreover easy to prevent, leaving only DPA as a serious menace to smartcard integrity. We show, however, that with careful experimental technique, SPA allows for extracting sensitive information easily, requiring only a single power-consumption graph. This even holds with respect to basic instructions such as register moves, which have previously not been considered critical. Our results suggest that SPA is an effective and easily implementable attack and, due to its simplicity, potentially a more serious threat than DPA in many real applications.

1 Introduction

It is the cryptanalyst's objective to obtain as much critical information as possible out of a cryptosystem, while keeping his effort and the risk of being detected at a minimum. In contrast to the design of a cryptographic algorithm, where security constitutes the central purpose, its ultimate physical implementation always depends on circuit implementation. Security aspects, as compared to efficiency, simplicity, or power consumption criteria, do still only play a marginal role in circuit design.

Kocher *et al.* [4] have proposed the following two kinds of so-called power-analysis attacks: simple power analysis (SPA), where the opponent tries to recover information about the secret key by simply measuring the power consumption of the computing device, and the more complex differential power analysis (DPA). Whereas the difficulty in SPA remains in the necessity for the attacker to know at which precise instant power consumption contains relevant information, DPA is more demanding in terms of the supplementary information needed. Above all, it requires a much larger number of experiments than does SPA.

In contrast to DPA, SPA merely requires the power consumption characteristics of one execution of the algorithm. However, SPA was previously considered

Ç.K. Koç and C. Paar (Eds.): CHES 2000, LNCS 1965, pp. 78–92, 2000.

unrealistic due to the destructive effect of noise deteriorating the measured signal. It was believed that, if anything at all, only conditional jump instructions testing for key bits might lead to successful SPA.

The results of our experiments, carried out on a smartcard-type microprocessor, stand in contrast to these beliefs and show that simple power analysis is an effective and easily implementable, hence serious, attack. This is even true for much simpler instructions than previously speculated, such as move operations, which cannot possibly be avoided by software countermeasures.

The outline of this paper is as follows. Section 2 provides an overview to the state of the art in power analysis and positions our results in this context (Section 2.4). In Section 3, we describe our experimental technique and the obtained results in detail. In Section 4, we draw the conclusions from the outcome of our experiments.

2 Power-Analysis Attacks to Cryptosystems

Power analysis is a physical attack to smartcard-based cryptosystems. It exploits the fact that the power dissipation of an electronic circuit depends on the actions performed in it. More specifically, the current flowing through the power lines of an operating microprocessor is dependent on the processed data. The following paragraphs describe and compare the types of power analysis that are currently examined. The hypotheses and results we discuss can be found in the recent publications [4], [6], [3].

2.1 Simple Power Analysis and Differential Power Analysis

Power analysis differs from most physical attack methods (see [1] and [2]) in many respects. First, it is not invasive and can thus be performed in a few instants; therefore, it can be used if a card-based action performed by an ordinary user is to be imitated by the eavesdropper, causing direct damage to the individual. Furthermore, the information side channel constituted by operation-related consumption can be accessed quite easily and without requiring a lot of specific knowledge about circuit or software implementation[1].

Those reasons make power analysis a type of attack which must be considered as a menace in case the eavesdropper is able to extract some information from the easily created side channel. Our objective has been to determine the amount of that information.

Kocher *et al.* [4] describe the two techniques which use the power-dissipation characteristics as a provider of side-channel information. Simple power analysis implies that the cryptanalyst measures the power consumption of the device operated during encryption or decryption, and evaluates the measured values (sampled at adequate instants, whose timing must be known or found by the

[1] Another question is how much information about the system is needed for exploiting the consumption information properly, but for now we just discuss accessibility of such a side channel.

attacker) directly, in order to correlate them with the key itself. On the other hand, differential power analysis is an attack which requires the availability of multiple power-consumption characteristics and ciphertexts out of a large number of diverse plaintext inputs.

The main advantage of DPA over SPA, apart from the fact that the attacker does not need to know implementational details of the target code (yet he must provide himself with all the other information necessary for performing DPA: a large number of ciphertexts and consumption graphs), is that the averaging process reduces the noise energy in the measured consumption signals. As the problem of extracting side-channel information from power-dissipation characteristics mainly lies in the many orders of magnitude between the absolute consumption values and the data-dependent differences between them, the influence of noise on SPA measurements can present an obstacle. Nevertheless, we could show that simple precautions in the measurement circuit, such as use of shielded cables and avoidance of ground loops, can raise the signal-to-noise ratio of the data obtained by SPA to an acceptable point.

The advantage of SPA over DPA is its low requirement in terms of amount of experiments and degree of device corruption. It certainly requires some insight into the structure of the implemented code, but extracting information about the program code with the help of microprobing tools is not a big obstacle for an experienced attacker (see [5] for details).

2.2 Physical Background of Simple Power Analysis

The power dissipation in CMOS cells such as logic gates, flip-flops, or latches mainly depends on changes of components' states rather than on the states themselves; e.g., for an inverter whose input voltage, applied to the connected gates of its cascaded PMOS and NMOS transistors, switches from high to low, the establishment of a transient short-circuit is induced. The rise of current in such a case is much larger than static dissipation. An in-depth analysis of short-circuit power consumption for a simple inverter cell is made in [7].

From these considerations, one might conclude that not the actual contents of the data bus, but rather the change in state of the internal registers from one instruction to the next would be measurable by power analysis. Nevertheless, our experiments enabled us to make both types of observations: the conductive properties of the data bus disclosed information about the absolute Hamming weight[2] of the transported data, and the rise in current induced by a change of state in internal registers was representative of the amount of bits that had changed in the data stored at this location, i.e., the transition counts[3]. Those two types of information are generated and retrieved independently, and combining them for cryptanalytic means is definitely interesting.

[2] The Hamming weight of a binary string is the number of ones that occur in it.

[3] The transition count between two consecutively processed data strings is the Hamming weight of their pointwise XOR-sum.

2.3 Previous Results on Simple Power Analysis

Messerges, Dabbish, and Sloan [6] specifically indicate the dangers of SPA and DPA on the DES encryption process. They provide an estimate on the reduction of the brute-force search space for the eight DES key bytes, for the case where the Hamming weights of these eight bytes are given, and also for the case where additionally the eight parity bits are known: without any supplementary information, there are 2^{56} possibilities, compared to about 2^{45} in the first and about 2^{38} in the second case.

They acknowledge that finding out the Hamming weights alone may be of little help, especially when larger keys than in DES are employed, but that this type of knowledge can get quite useful as soon as the key bytes are shifted, like during DES encryption.

In [6], the dangers are mentioned that may arise from knowledge of transition counts between key data and the data bus' contents previously to key data being transferred onto it. It is indicated that an attacker might easily find out what was written on the bus right before it was loaded with the crucial key byte, because this data is usually some fixed address or an instruction opcode. In this concern, our cryptanalytic methods – observing different but comparable processes (as are typical for execution of the encryption rounds), possibly separated by many instructions, and extracting the differences between them – take a different turn. What is proposed in [6] is an "instantaneous" analysis where real-time transitions are observed and evaluated. In [6], it is claimed that in such a case, the attacker requires some detailed knowledge about the source code of the algorithm's implementation (more precisely, that not only the code structure, but also the addresses of accessed registers and memory have to be known[4]).

The measurements exposed in [6] reflect the change in power dissipation when a bus which at first contains a memory address is loaded with various data values. Our results extend those measurements by showing that it is not necessary, nor at all helpful, to know storage addresses in order to find absolute Hamming weight values of data.

In our view, it must be proven that Hamming weight information for key bytes can be found by SPA. The two papers we discussed so far make this assumption and affirm that in principle "it can be done". Yet, they also claim that SPA is only possible for conditional branching instructions. Still, the corresponding quantitative results are not exposed. This triggered our desire to determine how effectively Hamming weights, and not merely transition counts, really can be found.

[4] We observe that those addresses could be generated randomly for every single smartcard (a kind of "fingerprint" addressing); to find out which addresses a certain card uses, it would thus be necessary to extract its specific source code – not always an easy task. It is more likely that the attacker is merely informed about the general structure of the code.

Biham and Shamir [3] propose a method which enables the attacker to identify the key-scheduling process during encryption by simple power measurements, when he has no access to information about algorithm implementation or timing. They show that the reasons for the vulnerability of DES, SAFER, and SAFER+ to this attack are uneven cyclic shifts and the way original key bits are grouped into subkey bytes. The authors found that during the key-scheduling process in DES, knowledge of the Hamming weights of the subkey bytes provides the key in a direct manner.

An important point is the implicit statement in [3] that SPA is an attack at least as dangerous as DPA if the cryptographic algorithm is designed in a way which makes it vulnerable to an attacker who has Hamming-weight information (and not just in the case where the implementor of the algorithm is not cautious about power analysis, and may create conditional jumps testing for key bits).

2.4 Our Results

We implemented the very simplest kind of power-analysis attack by observing the chip's power dissipation directly. Our main aim hereby was the extraction of information about data arguments of instructions. The method we employed to obtain maximum information from a microprocessor's power consumption characteristic was to compare a number of data-related processes identical except for one of the data or instruction properties we wished to examine (e.g., Hamming weight or Hamming-weight change, transition count, or absolute value of data or storage location; types of instructions or contents of instruction arguments; number of bits changing from high to low and inversely; more generally, the different types of changes that may take place). Then, we had to find out which of those properties could be at the origin of the observed variations.

In [4], it is claimed that SPA is easily made impossible by avoiding the use of key bits in conditional branching or jump instructions, whose dissipation characteristics distinguish themselves clearly from other operations. Yet, our results, obtained without involving conditional branching on sensitive data, indicate that even simple move instructions can reveal critical information.

Additionally, our results show that if the device is operated at sufficiently low frequency and high supply voltage[5], it is not even necessary to average noise out of the consumption characteristics in order to obtain key information. This implies that indeed a single experiment delivers enough insight to obtain key-relevant information.

Although DPA can represent a powerful attack on cryptosystems, as it directly aims at obtaining key bits, it is not necessarily a "very low cost" attack in the sense of easy feasibility. As the menace constituted by an attack is inversely proportional to the expense required for its performance, successful SPA should be regarded as especially dangerous due to its low cost.

[5] In an SPA scenario, operating frequency and supply voltage are considered to be under the control of the attacker.

3 Experimental Method

We chose the PIC16C84 chip [8] as the processor for our experiments, which were run at 4 MHz and 4.5 V supply voltage. This particular processor is similar in structure to most of the microprocessors in use for smartcard systems.

The method we used to investigate data dependency of the PIC's power dissipation was to design test routines in the processor's assembly language, making it perform certain instructions with varying data arguments. We then acquired the power-consumption characteristics generated during program execution; in the next step, we found "zones" of high correlation between data and consumption, which we further investigated. The conclusions drawn from these investigations constitute the results of our query.

3.1 Data Dependency in Move Instructions

In order to evaluate changes in power dissipation due to writing different data values into a certain memory location or register, we executed the following assembly-language program (see [8] for details) as an infinite loop:

```
; define registers VAL, PORTB, PORTA, REG:
VAL    equ 0x08
PORTA  equ 0x05
PORTB  equ 0x06          ; PORTA, PORTB: output ports
REG    equ 0x0c

start
      clrf    REG
      movlw   D'255'
      movwf   VAL        ; 0: move 255 to source value register

loopstart
      movfw   VAL, 0     ; 1: move new value to accumulator
      nop                ; 2
      nop                ; 3
      movwf   REG        ; 4: ! move value from accumulator
      nop                ; 5           to internal register !
      nop                ; 6
      movwf   PORTB      ; 7: move value to PORTB
      bsf     PORTA, 0   ; 8: set strobe bit (LSB of PORTA)
      bcf     PORTA, 0   ; 9: clear strobe bit
      clrf    PORTB      ;10: clear data in port B
      decfsz  VAL        ;11: decrease value,
      goto    loopstart  ;12           back to loopstart if !=0
      decf    VAL        ;13: set value to 255
      goto    start
```

This way, the numbers which are consecutively written into `REG` range from `255` to `0` (cf. instruction 4), decreasing by steps of one. We also examined variants of this program, where the transferred values were either increased by steps of 3, or decreased by steps of 1. Additionally, we ran the same program structure, replacing the `movwf` in instruction 4 by other commands combining moves and logical operations.

3.2 Finding Data Dependency

We acquired analog data representing the processor's power dissipation, sampled at 200 MHz (i.e., 50 samples per card cycle), by measuring the voltage over a probing resistor connected between the microprocessor's ground pin and the overall circuit's ground. We then wanted to find the instants during execution of the previously described program loops where data dependency of power consumption could be observed. At this point, assumptions were drawn as to what changes in which properties of data could induce measurable variations in power dissipation.

We investigated the data dependency of the acquired voltage samples in the following way: for every data value (as we examined an eight-bit processor, those values range from 0 to 255), one loop of the test program was run. Data dependency is likely to occur at several instants, e.g., those where data is written to the output ports of the processor; yet, this process is much less interesting than the write operation triggered by the `movwf`-instruction, which transfers a value from the accumulator to one of the internal registers, or inversely. In order to see clearly at what exact instant during the execution of this command the dependency between power dissipation and data is maximal, a correlation factor (correlating the measured values to the investigated data properties) was computed for every list of length 256, containing a measure of power dissipation at a certain stage of the loop execution for every data value. Thus, for every sampling moment k (the range of k is dependent on the number of assembly instructions per loop and sampling rate) during the loop execution, there exists a list v_k,

$$v_k = [v_k(0), v_k(1), \ldots, v_k(255)],$$

where $v_k(j)$ is the voltage measured over the probing resistor at the moment of the k^{th} sample during execution of the loop with data argument j. We are now interested in correlations between $v_k(j)$ and certain properties $p(j)$ of the data j for fixed k.

There are various ways to compute correlations between two quantities. For instance, one might be in the situation of wanting to evaluate the degree of correlation between two sets of data samples with an unknown joint distribution. However, in the present case, numerous data sets are compared against one another, so the relative rather than the absolute value of correlation is interesting; we were primarily interested in detection of local maxima. Therefore, setting the average current consumption at the moment of the k^{th} sample and the average of the investigated data property over all data arguments to $\overline{v_k}$ and $\overline{p}$ respectively,

we simply computed the Pearson correlation factor r_k,

$$r_k = \frac{\sum_j (v_k(j) - \overline{v_k}) \cdot (p(j) - \overline{p})}{\sqrt{\sum_j (v_k(j) - \overline{v_k})^2} \cdot \sqrt{\sum_j (p(j) - \overline{p})^2}}$$

and compared the correlation factors for different moments k during program loop execution.

In accordance with our expectations, we found that correlation between power dissipation and Hamming weight of processed data indeed occurs. This fact can be stated after inspection of the different correlation graphs which have been drawn with respect to direct data, Hamming weight, and transition counts. The second type of correlation graph contains valuable information in the sense of clear peaks indicating high correlation during the `movwf`-instruction. In addition, inspecting the transition-counts correlation graph we found that in the same instruction, there is also an instant where power consumption is proportional to the number of bits that were inverted from the previously transferred data value to the current one. But we may not yet be led to the conclusion that Hamming weights and transition counts are the only data properties correlated to consumption; it is always possible that we "overlooked" certain other kinds of correlation.

A typical graph of correlation values with respect to Hamming weight is shown in Figure 1. From the correlation graph, we extracted local peaks, indicating that "something interesting" might be happening at the k^{th} instant during loop execution. We then inspected the corresponding v_k and evaluated if indeed data dependency could be observed. Examples of typical v_k's, extracted in the described manner, are given in Figures 2-10.

3.3 Noise Level of the Acquired Signals

Figures 2-10 indicate the striking similarities between the components of v_k and the Hamming weights (Fig. 2-4, 7-10) or the transitions counts (Fig. 5, 6) of the sequence of processed data. Yet, the visualization of this similarity is just an intuitive hint that Hamming information is leaked; the noise level of the obtained signal still had to be examined.

Thus, we now evaluated whether the quality of the extracted measurement data was at all high enough for making assumptions about Hamming weights of the processed instruction arguments. In the given case, this evaluation primarily consisted of the question whether the data consumption values could be grouped in a unique manner, so that they would form clusters of points which could be assigned a single Hamming weight, and whether the noise induced by the measurement was low enough in order to make those attributions in a correct manner.

We made the separation into nine clusters of points and observed that the averages of every cluster are separated by voltages of about $\Delta V \approx 5\,\text{mV}$. Those cluster distances remain constant for all Hamming-weight values except for zero,

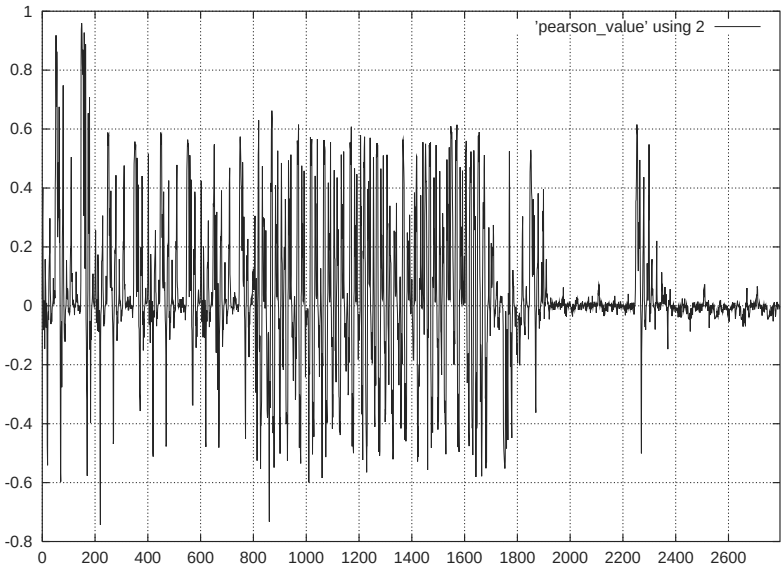

Fig. 1. Pearson correlation factors during program loop execution. Variation of correlation factors can be observed during the last five loop instructions: two nop's, movwf[data], two nop's (200 samples per instruction).

which induces much lower consumption. Thus, the maximum admitted noise level is $n_{max} = \Delta V/2$; every noise contribution higher than this will lead to an erroneous conclusion about the Hamming weight of processed data.

In our experiments, we were indeed able to locate v_k's where Hamming weight attribution could be done in an unequivocal manner. For a real attacker things are different: unless he already knows the timing of the investigated process, he cannot find the proper instant k without the help of correlations. Yet, we found that power dissipation at this crucial instant (where best indication of Hamming weights is given) is characterized by maximal correlation between current consumption and data, minimal distortion of power consumption by processes other that the loading of key bytes on the internal data bus, and minimum variance among the clusters of consumption samples. Thus, even if the attacker is a priori unable to locate the desired instant k, he might reach this aim by using those properties of the acquired data.

4 Concluding Remarks

We have shown that SPA can be done with extremely simple infrastructure and adequate experimental technique. Even basic assembly instructions such as register moves provide information about Hamming weights of on-bus data *and* transition counts between data items written into memory locations or registers.

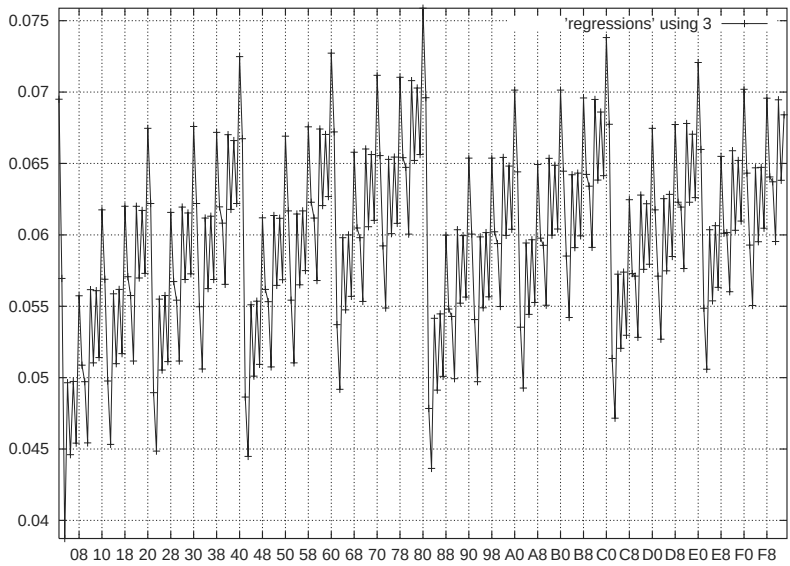

Fig. 2. Voltage values for 256 loop executions. Here, v_k is extracted at the instant of highest correlation between Hamming weight of the data sequence and power dissipation (during instruction 4). x-axis: data values j; y-axis: $v_k(j)$ in [V]. A zoom-in is shown in Figure 3.

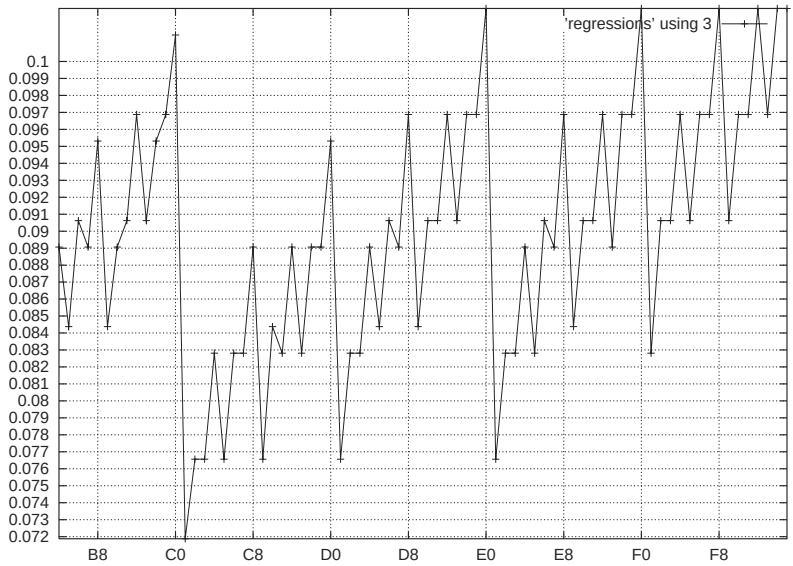

Fig. 3. The values $v_k(j)$ for j ranging from 180 to 256, increasing by steps of 1.

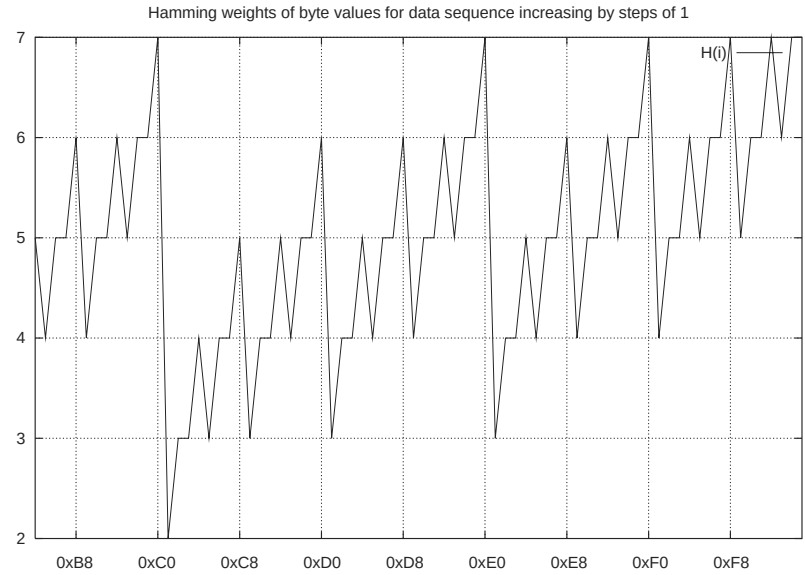

Fig. 4. For comparison, this figure gives the computed Hamming weights of the data processed in the investigated loops.

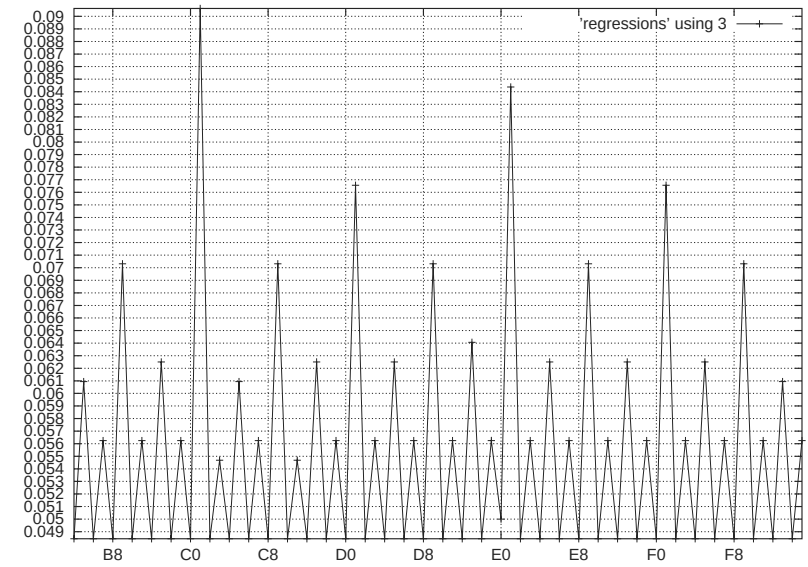

Fig. 5. Measured voltage values at instants of highest correlation between transition counts and power consumption, for j going from 180 to 255, increasing by steps of 1.

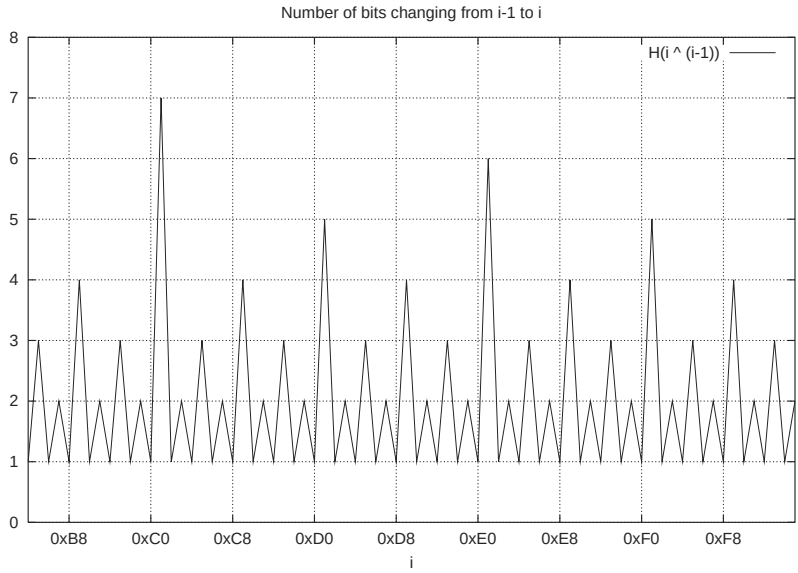

Fig. 6. Computed transition counts for data going from 180 to 255, increasing by steps of 1.

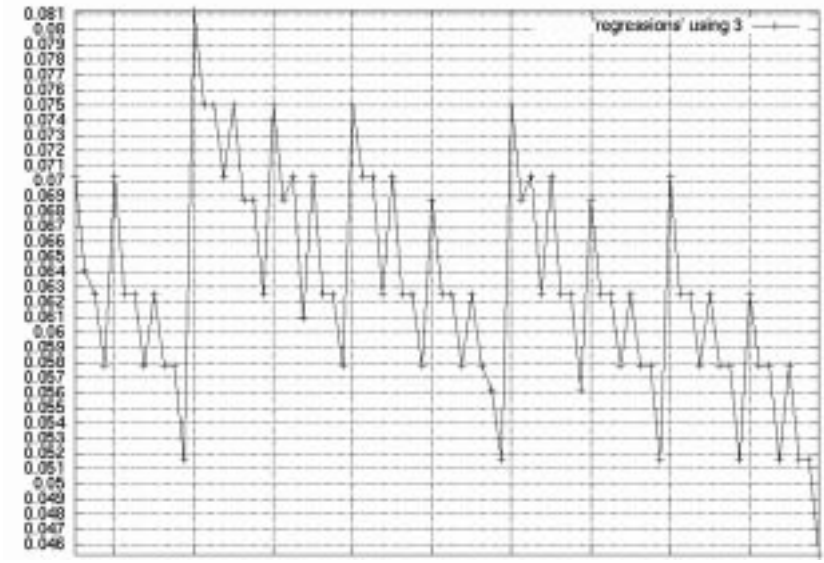

Fig. 7. The values $v_k(j)$ for data j transferred by the move-instruction ranging from 76 down to 0, decreasing by steps of 1.

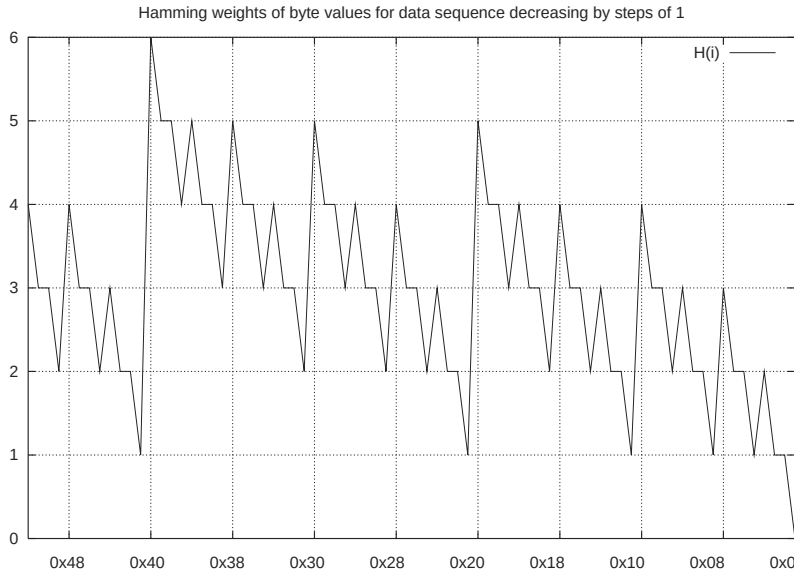

Fig. 8. Computed Hamming weights for data ranging from 76 down to 0, decreasing by steps of 1.

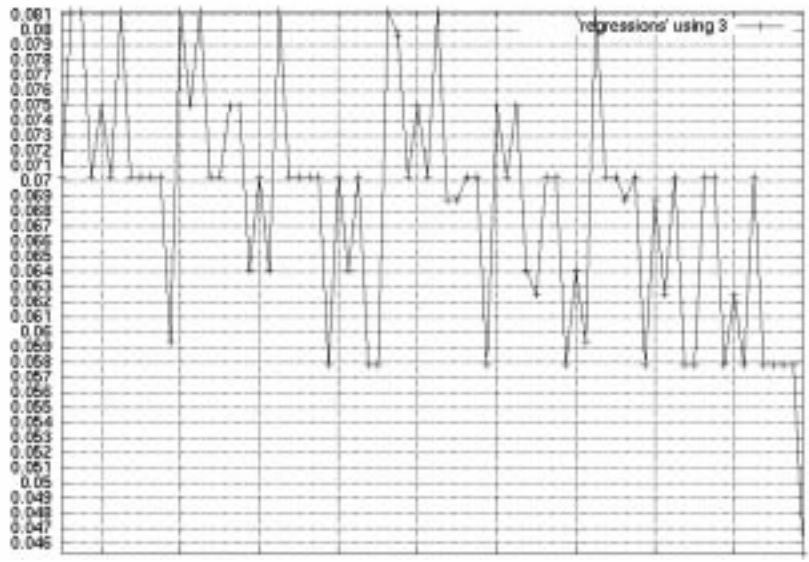

Fig. 9. Measured voltage values for transferred data j ranging from 228 to 0, decreasing by steps of 3.

Fig. 10. Computed Hamming weights for data ranging from 228 to 0, decreasing by steps of 3.

The SPA attacker does not require conditional jumps on sensitive data in order to obtain this information, contrary to what was supposed until now. Using appropriate noise shielding, we could show that a single experiment suffices in order to draw the desired key information from the power consumption of a smartcard processor performing cryptographic operations.

When examining instructions other than `movwf`, we have observed that the extracted values sequence is identical, and hence independent of the instruction type; thus, `iorwf, xorwf, rrf, subwf` all yield resembling v_k's when correlated to Hamming weight and transition count. In order to explain this, we indicate that the "crude" data is transferred over the internal bus before getting involved with mathematical or logical operations in the ALU. This data transfer is, at certain instants, the sole reason for characteristic power consumption values; whatever takes place inside the arithmetic and logic unit of course causes data- and operation-dependent power dissipation, but it is not easily possible nor at all necessary to analyze the power consumption of this type of activity.

It was our objective to find out whether Hamming weights and transition counts are really yielded by SPA, and the question can be answered by a clear yes. Even if we assume that nothing but what we found can be found at all, this still is a menace to smartcard holders' security if the attacker is able to synchronize with the implemented software.

Acknowledgments

This research was carried out during the author's stay with the Security Group, Computer Lab of Cambridge University, UK. The author would like to thank Ross Anderson, Stefan Wolf, Markus Kuhn, Anna Lysyanskaya, and Hubert Kaeslin for stimulating discussions and proofreading, and the anonymous referees for helpful comments. This project was supported by NDS Technologies Israel Ltd.

References

1. R. J. Anderson and M. G. Kuhn, Low Cost Attacks on Tamper Resistant Devices, *5th International Workshop on Security Protocols*, 1997.

2. R. J. Anderson and M. G. Kuhn, Tamper Resistance - a Cautionary Note, *Proceedings of the Second USENIX Workshop on Electronic Commerce*, 1996.
3. E. Biham and A. Shamir, Power Analysis of Key Scheduling of the AES Candidates, *Proceedings of the Second AES Candidate Conference*, 1999.
4. P. Kocher, J. Jaffe, and B. Jun, Differential Power Analysis, *Advances in Cryptology – Proceedings of CRYPTO '99*, Lecture Notes in Computer Science, Vol. 1666, Springer-Verlag, 1999.
5. M. Kuhn and O. Kömmerling, Design Principles for Tamper-Resistant Smartcard Processors, *Proceedings of the USENIX Workshop on Smartcard Technology*, 1999.
6. T. S. Messerges, E. A. Dabbish, and R. H. Sloan, Investigations of Power Analysis Attacks on Smartcards, *Proceedings of the USENIX Workshop on Smartcard Technology*, 1999.
7. H. J. M. Veendrick, Short-Circuit Dissipation of Static CMOS Circuitry, *IEEE Journal of Solid-State Circuits*, Vol. SC-19, No.4, 1984
8. Datasheet of the PIC16C84 Processor, available at `http://www.microchip.com`.